REDACTED VERSION

Exhibit A1
to
C. Cramer Declaration

Karma M. Giulianelli (SBN 184175)
BARTLIT BECK LLP
1801 Wewatta St., Suite 1200
Denver, CO 80202
Telephone: (303) 592-3100
Facsimile: (303) 592-3140
karma.giulianelli@bartlitbeck.com

Hae Sung Nam (*pro hac vice*)
KAPLAN FOX & KILSHEIMER LLP
850 Third Avenue
New York, NY 10022
Telephone: (212) 687-1980
Facsimile: (212) 687-7715
hnam@kaplanfox.com

Interim Co-Lead Counsel for the Proposed Classes

[Additional Counsel on Signature Page]

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

IN RE GOOGLE PLAY CONSUMER ANTITRUST LITIGATION

RELATED ACTIONS:

Epic Games Inc. v. Google LLC et al., Case No. 3:20-cv-05671-JD

In re Google Play Developer Antitrust Litigation, Case No. 3:20-cv-05792-JD

State of Utah, et al., v. Google LLC, et al., Case No. 3:21-cv-05227-JD

Match Group, LLC, et al. v. Google LLC, et al., Case No. 3:22-cv-02746-JD

No. 3:20-CV-05761-JD

CONSUMER PLAINTIFFS' NOTICE OF MOTION, MOTION FOR CLASS CERTIFICATION, AND MEMORANDUM IN SUPPORT

Hearing Date: August 4, 2022
Hearing Time: 10:00 a.m.
Courtroom: Courtroom 11, 19th Floor
Judge: The Honorable James Donato

NOTICE OF MOTION AND MOTION FOR CLASS CERITIFICATION

PLEASE TAKE NOTICE that on August 4, 2022, at 10:00 a.m., before the Honorable James Donato, of the United States District Court of the Northern District of California, San Francisco Division, 450 Golden Gate Avenue, San Francisco, California, Courtroom 11, 19th Floor, Plaintiffs Mary Carr, Daniel Egerter, Zack Palmer, Serina Moglia, Matthew Atkinson, and Alex Iwamoto, on behalf of themselves and all others similarly situated, will and do now move the Court for an order granting Plaintiffs' Motion for Class Certification pursuant to Federal Rules of Civil Procedure 23.

Plaintiffs seek entry of an order: (1) certifying a proposed Rule 23(b)(3) class; (2) certifying a proposed Rule 23(b)(2) class; (3) appointing Plaintiffs Mary Carr, Daniel Egerter, Zack Palmer, Serina Moglia, Matthew Atkinson, and Alex Iwamoto as representatives of the classes; and (4) appointing Karma M. Giulianelli of Bartlit Beck LLP and Hae Sung Nam of Kaplan Fox & Kilsheimer LLP as Co-Lead Class Counsel for the classes. Plaintiffs propose that the classes for their Sherman Act Sections 1 and 2 claims (Counts 1-6) as well as their Cartwright Act (Counts 7-10) and Unfair Competition (Count 11) claims, be defined as follows:

<u>RULE 23(b)(3) MULTISTATE DAMAGES CLASS</u>:

All persons in the following U.S. states and territories:

> Alabama, Georgia, Hawaii, Illinois, Kansas, Maine, Michigan, Ohio, Pennsylvania, South Carolina, Wisconsin, Wyoming, American Samoa, Guam, Northern Mariana Islands, Puerto Rico, and the U.S. Virgin Islands

who paid for an app through the Google Play Store[1] or paid for in-app digital content (including subscriptions or ad-free versions of apps) through Google Play Billing on or after August 16, 2016, to the present.

<u>RULE 23(b)(2) MULTISTATE INJUNCTIVE RELIEF CLASS</u>:

All persons in the following U.S. states and territories:

> Alabama, Georgia, Hawaii, Illinois, Kansas, Maine, Michigan, Ohio, Pennsylvania, South Carolina, Wisconsin, Wyoming, American Samoa, Guam, Northern Mariana Islands, Puerto Rico, and the U.S. Virgin Islands

who currently own a mobile phone or tablet with an authorized and preinstalled version of Google's Android OS capable of accessing the Google Play Store.

---

[1] Capitalized terms "Google Play Store," "Google Play Billing," and "Defendants" are used in the same sense as defined in the operative consumer Complaint. ECF No. 241.

> Excluded from both Classes are Defendants and their officers, directors, employees, and successors; any person or entity who has (or had during the class period) a controlling interest in any Defendant; any affiliate, legal representative, heir, or assign of any Defendant and any person acting on behalf of any Defendant; any judicial officer presiding over this action and the members of those officers' immediate families and judicial staffs; all governments and their agencies; and any juror assigned to this action.

This motion is based upon this Notice of Motion, the accompanying Memorandum of Points and Authorities, all filed supportive declarations and exhibits, the expert reports of Dr. Hal Singer and Dr. Douglas Schmidt, the records on file in this action, and any argument that may be presented at or before the hearing on this Motion.

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Cases

Other Authorities

Rules

MEMORANDUM OF POINTS AND AUTHORITIES

For years, Google has engaged in exclusionary practices that have allowed it to obtain a dominant position in the market for distributing Android mobile device applications (apps) and to thwart rival app stores from emerging. Google uses its power to inflate the prices of Android apps by taking, with limited exceptions, a 30% slice of every app sale through the Google Play Store. Google has also unlawfully extended its monopoly power in the Android App Distribution Market to the In-App Aftermarket through its requirement that developers use Google's billing services for all in-app sales and its prohibition on steering to alternate providers—allowing Google to take a cut in perpetuity of every subsequent purchase of digital content in each app.[2]

Internal Google documents reveal that its meteoric rise against such titans as Amazon, Samsung, AT&T, T-Mobile, Motorola, and Verizon was the product of a calculated, multifaceted strategy to eliminate competition. That strategy included an array of conduct that courts have long found to be exclusionary—bribes, deception, contractual restrictions, and pretextual technological barriers—which secured and maintain the Play Store's insurmountable lead over potential competitors.

The Consumer Plaintiffs purchased apps from the Play Store or made in-app purchases through Google Play Billing. Consumers paid Google directly for these purchases, and, accordingly, are direct purchasers who may sue Google under federal antitrust laws. *Apple Inc. v. Pepper*, 139 S. Ct. 1514, 1520 (2019). But for Google's anticompetitive conduct, Plaintiffs and class members would have paid lower prices for apps and in-app purchases and would have benefitted from expanded choice. Plaintiffs seek damages from and injunctive relief against Google, and therefore seek certification of both a Rule 23(b)(3) damages class of consumers and a (b)(2) injunctive relief class of current owners of devices operating with an authorized version of Google's Android OS.[3]

[2] "Android App Distribution Market" and "In-App Aftermarket" are defined in the Class Certification Report of Hal J. Singer, Ph.D. Ex. 2 (Singer Rpt.) ¶¶ 2, 22-32. All exhibit citations are citations to the exhibits to the Giulianelli Declaration, filed herewith.

[3] As used in this memorandum, "Android OS" refers to Google's licensed version of Android, as opposed to other versions of the freely available "open-source" version. *See* Ex. 4 (Expert Report of D. Schmidt) at 8-9.

Plaintiffs' claims depend upon central, common questions of fact and law, each of which focuses on Google's conduct and is capable of classwide resolution. They include: (1) whether Google has monopoly power in the Android App Distribution Market and the In-App Aftermarket; (2) whether Google's contractual restrictions on manufacturers, carriers, and developers are unreasonable restraints of trade; and (3) whether Google's practices impacted class members and resulted in supra-competitive prices paid by consumers. With respect to the last issue, Plaintiffs have developed expert economic evidence capable of demonstrating that Google did indeed cause classwide antitrust injury. That evidence will show that virtually all members of the proposed class were injured by Google's "coercive activity," "prevent[ing] its victims from making free choices between market alternatives," which constitutes cognizable antitrust injury. *Ellis v. Salt River Project Agric. Improvement & Power Dist.*, 24 F.4th 1262, 1274 (9th Cir. 2022). That evidence also includes a methodology for calculating aggregate damages to the class, as well as class members' individual damages.

Because common questions about Google's conduct will be the focus at trial, Plaintiffs satisfy Rule 23(b)(3)'s requirement that common questions "predominate over any questions affecting only individual members." None of the differences among class members (for example, which specific purchases were made) are of the kind that defeat class certification. *Olean Wholesale Grocery Coop., Inc. v. Bumble Bee Foods, LLC*, 31 F.4th 651, 678 (9th Cir. Apr. 8, 2022) (en banc) (common impact can be found "even when the market involves diversity in products, marketing, and prices"); *Tyson Foods, Inc. v. Bouaphakeo*, 577 U.S. 442, 453 (2016) (certification is proper where common issues predominate even where "other important matters will have to be tried separately, such as damages or some affirmative defenses peculiar to some individual class members"). Certification of an injunctive relief class is appropriate because Google "has acted or refused to act on grounds that apply generally to the class." Fed. R. Civ. P. 23(b)(2).

As explained in further detail below, the Court should certify classes under both Rule 23(b)(3) and (b)(2), appoint Plaintiffs as class representatives, and appoint Karma M. Giulianelli of Bartlit Beck LLP and Hae Sung Nam of Kaplan Fox & Kilsheimer LLP as co-lead class counsel.

STATEMENT OF COMMON FACTS

I. THE PROPOSED CLASSES AND THE JOINT PROSECUTION AGREEMENT WITH THE STATES

The classes Plaintiffs propose differ from those pleaded.[4] The proposed Rule 23(b)(3) class is limited to seventeen states and territories in lieu of the nationwide class pleaded because Plaintiffs have entered a Cooperation and Joint Prosecution Agreement (the "Joint Prosecution Agreement") with Attorneys General who have brought *parens patriae* claims. *See* Ex. 1. To pursue consumers' claims against Google most effectively and efficiently, Plaintiffs' Counsel and the thirty-nine Attorneys General asserting *parens patriae* claims in this case agreed in the Joint Prosecution Agreement that class certification would be sought only for consumers in states, districts and territories that have not asserted a *parens patriae* claim in this action.[5] Plaintiffs' Counsel and the Attorneys General also agreed to continue working jointly for the benefit of *all* U.S. consumers.

A second difference in the proposed classes is that Plaintiffs do not seek certification of a "repealer states" class. Because Google has consistently included a choice-of-law provision in its user agreements designating California law as controlling in litigation brought by users, California law governs the state law claims of all class members, regardless of where they reside and regardless of whether a particular state has "repealed" *Illinois Brick*.

Plaintiffs have also proposed a Rule 23(b)(2) injunctive relief class in part based on the Supreme Court's recognition that "individualized monetary claims belong in Rule 23(b)(3)," while "Rule 23(b)(2) applies only when a single injunction or declaratory judgment would provide relief to each member of the class." *Wal-Mart Stores, Inc. v. Dukes*, 564 U.S. 338, 360 (2011). The proposed (b)(2) class does not completely overlap with the proposed (b)(3) class, in that the former is limited to current Android users, while the latter includes anyone who was an Android user who purchased an app or in-app digital content using Google Play Billing during the class period.

[4] *See*, *e.g.*, *In re TFT-LCD (Flat Panel) Antitrust Litig.*, 267 F.R.D. 583 (N.D. Cal. 2010) (permitting adjustments to class definition at the class certification stage); *Castellar v. Mayorkas*, No. 17-cv-00491-BAS-AHG, 2021 U.S. Dist. LEXIS 170342 (S.D. Cal. Sept. 8, 2021) ("The definition of the class at the class certification stage may diverge from that set forth in the Complaint ....").

[5] If for any reason a State cannot or does not pursue its *parens patriae* claims, Rule 23 provides the necessary flexibility to expand a class to include residents of that State, as Rule 23 permits the Court to modify a certified class.

II. COMMON CLASSWIDE EVIDENCE OF THE RELEVANT MARKET

The relevant antitrust markets in this case will be defined by common evidence. Virtually all mobile devices throughout the world (more than 99%) use either Apple's iOS, exclusive to Apple devices, or Google's Android OS, licensed to multiple Original Equipment Manufacturers ("OEMs"), giving it over 99% of the smartphone market for licensed mobile operating systems. Ex. 2 (Singer Rpt.) at ¶ 37. Once a consumer buys an Apple or Android device, she has no choice but to use the app stores that are available for that device, meaning there is no direct competition between Apple and Google for a consumer's app or in-app business. Ex. 4 (Schmidt Rpt.) at 5-6 (explaining incompatibility of mobile operating systems). Google maintains near complete control over the relevant Android App Distribution Market, and, because of its restrictions, over the In-App Aftermarket for services used to transact the sale of in-app content. Google's expert accepts these market definitions for class certification purposes. S*ee* Ex. 5 (Burtis Rpt.) ¶ 43.

III. COMMON CLASSWIDE EVIDENCE OF GOOGLE'S UNLAWFUL CONDUCT

Common evidence related to Google's anticompetitive conduct will be the main focus of any trial. Plaintiffs will prove through common evidence that Google's dominance was acquired in both the Android App Distribution Market and In-App Aftermarket by anticompetitive means. Common evidence will show Google unlawfully: (1) paid carriers rents too high for a competitor to profitably enter; (2) imposed contractual restrictions on OEMs; (3) prohibited developers from steering their customers to competitors; (4) bribed major developers; (5) erected overly restrictive and pretextual technological barriers and misleading warnings to deter consumers from downloading apps outside the Play Store; (6) persuaded entrants not to compete; and (7) leveraged its power into the separate in-app services market by tying its separate Google Play Billing product to every in-app sale of digital content.

First, Google paid billions of dollars to mobile carriers, making it unpalatable for them to compete with Google's app store. Initially, Google deliberately lost money on each app transaction, pushing out carriers and third parties from profitably participating in carrier distribution channels. In the early days of Android smartphones, Google appreciated that "the greatest threat to Android Market is that carriers can easily set up their own, controlled, application market." Ex. 6

(GOOG-PLAY-001423609). Google's solution to neutralize the threat was "giving generous revenue share that more or less matches what they would make from their own markets," until a point where "carriers will be unable to compete with their own offerings because their own offerings will be so limited in comparison." *Id*. To enact this plan, Google initially gave 70% of revenue to app developers, 25% to carriers, and kept only the remaining 5% for itself. *See, e.g.*, Ex. 7 (GOOG-PLAY-001385324) at -345. Google's cut was small enough that it "purposefully" took a loss from each transaction and told developers that "Google will not be operating the Android Market as a profit center. … Google will collect a small charge to cover costs of handling and billing." Ex. 8, (GOOG-PLAY-005564421); Ex. 9 (GOOG-PLAY-001075142) at -143. Google recognized that taking a loss by giving away 95% of each transaction effectively prevented entrants from competing profitably. Ex. 10, (GOOG-PLAY-001547487) at -488 ("How do you think they are going to get distribution when we give the carriers 30%").

By 2013, Google's revenue-sharing agreements with carriers had achieved their intended effect, as Google Vice President Jamie Rosenberg later acknowledged: "We cut carriers in to disincentivize building their own stores and fragmenting the ecosystem. It worked." Ex. 11 (GOOG-PLAY-000439987.R) at -012.R. As Google recognized, giving up control over app distribution was "a bitter pill," for the carriers, but Google's "generous rev share" was the "sugar" that made it "go down." Ex. 12 (GOOG-PLAY-005559390.R) at -395.R. With its monopoly solidified, Google knew it could "change the rules/get a better deal" by lowering carriers' share of revenues. Ex. 13 (GOOG-PLAY- 001337211) at -226. After it had sidelined all the major, would-be carrier and non-carrier competitors for Android app distribution, Google effectively ended revenue sharing in all but limited circumstances. [6] It did not, however, reduce its take or pass those savings to its customers, consistent with its public promise that "Google does not take a percentage" of the revenues from app distribution, but, instead, kept the entire 30% for itself.[7] By increasing its take

---

[6] Google first eliminated credit card sharing with T-Mobile USA and Sprint, Ex. 14 (GP MDL-TMO-0001831) at -838; Ex. 15 (GP MDL-TMO-0002071) at -094; then AT&T, Ex. 16 (GOOG-PLAY-003604601) at -603; and Verizon. Ex. 17 (GOOG-PLAY-003604896) at -910; Ex. 18 (GOOG-PLAY-003605103) at -106; Ex. 19 (GOOG-PLAY4-002178046) at -049.

[7] Ex. 21, ("Android Market: Now available for users", published on October 22, 2008, https://android-developers.googleblog.com/2008/10/androidmarket-now-available-for-users.html).

from 5% to 30% of each transaction, Google quickly began enjoying supra-competitive margins. Ex. 20 (GOOG-PLAY-000445443.R) at -461.R.

Second, Google has imposed a variety of exclusionary contractual restraints on OEMs of mobile devices. Although Google offers many of its own apps for "free," Google requires any OEM that wishes to install even a single Google app to preinstall a bundle of Google apps called Google Mobile Services ("GMS"), including Search, Maps, Chrome, Gmail, YouTube, and the Play Store. Ex. 22 (GOOG-PLAY-003776161.R) at -177.R (manufacturers "cannot pick apps a la carte"). GMS not only contains Google's most popular apps, but also application programming interfaces ("APIs") necessary for the majority of third-party apps to work. Ex. 4 (Schmidt Rpt.) at 10-12. OEMs therefore have no commercially viable alternative but to license GMS. To gain access to those critical APIs, however, an OEM must enter into a Mobile Application Distribution Agreement ("MADA") in which Google requires the OEM to preinstall the Play Store on its devices and to prominently display it on the home screen.[8] Google knows the importance of prominent placement, observing internally, "If we were honest we would admit that most users and developers aren't consciously 'choosing,' they are going with the default." Ex. 25 (GOOG-PLAY-000292207.R) at -226.R; Ex. 26 (GOOG-PLAY-000832471); Ex. 27 (GOOG-PLAY-006355073) ("Fortunately, we'll always have the placement/pre-install advantage which is 90% of the battle.").

On top of the MADA requirements, Google has recently—in the face of agitation from large developers and potential market entrants—begun negotiating agreements with OEMs securing *exclusivity* for the Play Store. Ex. 28 (GOOG-PLAY-000443763.R) at -773.R. Google recently entered into new revenue sharing agreements, known as "RSA 3.0," with at least [redacted] covering an estimated [redacted] shipped devices in 2020 and 2021. Ex. 2 (Singer Rpt.) ¶ 117. If a manufacturer agrees not to preinstall any Play Store competitors, Google pays that manufacturer [redacted] revenue for that device. Ex. 28 (GOOG-PLAY-000443763.R) at -775.R. Google expected to spend [redacted] to secure these deals, with a stated goal to [redacted] Ex. 29 (GOOG-PLAY4-

[8] Illustrative examples include Motorola's 2018 MADA and Samsung's 2017 MADA. Ex. 23 (GOOG-PLAY-000808375); Ex. 24 (GOOG-PLAY-004552342); *see also* Ex. 2 (Singer Rpt.) at 45 ¶ 97 (discussing preinstallation and prominent displacement requirements).

007239946). By May 2020, Google executives declared that the RSA 3.0 program had successfully eliminated the "risk of app developer contagion" by preventing OEMs from preloading any app store other than Google Play. Ex. 30 (GOOG-PLAY-000558461.R) at -465.R, -466.R. These agreements alone substantially foreclose competition.

Third, Google erected roadblocks between the software developers who write Android apps and the consumers who buy them, effectively preventing the two sides from doing business outside the Play Store. In what it previously called a "non-compete" provision in its agreements with app developers, Google's Developer Distribution Agreements ("DDAs")—which developers must execute to offer their apps on the Play Store—prohibit developers from steering customers to lower-priced app stores or websites to download apps or buy in-app content. Ex. 31 (GOOG-PLAY-000225435) ("it's against policy to direct users to content outside of the Play Store"). They also forbid developers from using the Play Store "to distribute or make available any Product that has a purpose that facilitates the distribution of software applications and games for use on Android devices outside of Google Play."[9] Google's Payments Policy forbids developers from using alternative payment processors. Ex. 34 (Google Play Payments Policy, https://support.google.com/googleplay/android-developer/answer/9858738). These contractual provisions, which are not justified by any technical or security reason, restrain competition for transacting the purchase of in-app content.

And Google imposes what is essentially a gag order on the free flow of information. In order for a developer to distribute an app through the Play Store, the developer must agree not to use consumer information it obtains through the sale of its apps or in-app content in the Play Store to direct the consumer to purchase apps or in-app content outside the Play Store. The DDA thus *permanently* precludes developers from using their own customers' information to steer elsewhere. Ex. 35, (GOOG-PLAY-000053875) at -876 § 4.9.

Fourth, Google paid top developers to prevent competitors from gaining an edge. By 2018, some larger developers, fed up by Google's supra-competitive fees, threatened to defect to Samsung or start their own app stores. Ex. 36 (GOOG-PLAY-003332817.R) at -823.R-824.R. At the

[9] Google first called this a "Non-Compete" provision. Ex. 32 (GOOG-PLAY-000054021) at -022. By 2014, Google renamed it "Alternative Stores." Ex. 33 (GOOG-PLAY-000054039) at -041.

same time, Google learned that Samsung was "actively reaching out to top game developers and asking them [to] consider distribution through Samsung Galaxy App store first or instead of via Play," which would have allowed Samsung to compete more effectively. Ex. 37 (GOOG-PLAY-004509271) at -272; Ex. 36 (GOOG-PLAY-003332817.R) at -835.R. Fearing "contagion," Google hatched a plan and gave it a secret internal codename: "Project Bear Hug." Ex. 38 (GOOG-PLAY-000560173.R) at -178.R-179.R. Google estimated that defection of the small handful of developers large enough to pose a threat could cost Google [redacted] **revenue /** [redacted] margin impact annually in three years ([redacted] revenue /[redacted] margin risk cumulative 2019-2022)." Ex. 39 (GOOG-PLAY-000000807) at -808 (bold in original). Because Google's 30% take was [redacted] the Project Bear Hug payments came in the form of [redacted] In return, Google required developers to [redacted] As a Google employee characterized it, Google was "mitigating our risk of losing out to competition" from Samsung and other competitors. Ex. 43 (Koh Dep. Tr.) at 362:22-363:1, 364:13-365:4, 367:4-16. Project Bear Hug worked. After Google signed up the largest developers, [redacted] Ex. 43 (Koh Dep. Tr.) at 368:4-369:4; Ex. 44, (GOOG-PLAY-004146689.R) at -697.R [redacted]

Fifth, Google has deliberately created unnecessary technical obstacles to prevent consumers from purchasing apps and in-app content outside the Play Store. Google has purposely made it difficult for consumers to download apps from rival app stores, requiring consumers to navigate a cumbersome "sideloading" process. Ex. 4 (Schmidt Rpt.) at 22-23, 27-35. For example, at one point, installation of Amazon Underground on an Android device required consumers to navigate a complicated 19-step process. Ex. 45 (GOOG-PLAY-000297309.R) at -310.R-314.R. When a

---

[10] Ex. 40 (GOOG-PLAY-004588725) at -726 (noting that [redacted]; Ex. 41 (GOOG-PLAY-000559379.R) at 382.R, 384.R; Ex. 39 (GOOG-PLAY-000000807); Ex. 42 (GOOG-PLAY-000229696).

[11] [redacted] *See* Ex. 39 (GOOG-PLAY-000000807) at -810.

user attempts to change the default security setting that blocks downloads of apps outside the Play Store, including alternative app stores, Android displays misleading warnings, such as that changing settings would make "your phone and personal data … more vulnerable to attack." Ex. 4, (Schmidt Rpt.) at 28-31. Android uses onerous download warnings even for reputable companies and would-be competitors like Amazon, terming it an "unknown source" even though less burdensome mechanisms exist to validate the safety of the app. *Id.* at 37-41. Internally, Google refers to these extra steps and warnings as "friction," Ex. 4 (Schmidt Rpt.) at 32-33, in recognition that the steps stop consumers from and deny competitors the ability to provide the same one-click service Google can provide.

Sixth**,** Google has repeatedly pressured competitors and would-be competitors not to compete. Over the past several years, Google perceived Samsung's Galaxy Store as a potential competitor to the Play Store. Ex. 46 (GOOG-PLAY-004253884) at -907. To prevent competition from Samsung, Google met with Samsung on numerous occasions and offered a variety of financial incentives not to compete with the Play Store, aiming to prevent Samsung from "duplicat[ing] our services on Android." Ex. 47 (GOOG-PLAY-001449339). Google attempted to persuade Samsung to essentially exit the app store business altogether. As Google put it to Samsung, it didn't "believe Samsung should be cultivating its own developer ecosystem." Ex. 48 (GOOG-PLAY-006359924). Specifically, in 2019, Google Play's leadership offered Samsung a bespoke deal for an ongoing investment of tens of millions of dollars with the stated goal of "[p]revent[ing] unnecessary competition on store." Ex. 49 (GOOG-PLAY4-004259430). In these discussions, Google told Samsung that it could "solve for [Samsung's] goals without competing commercially," Ex. 50 (GOOG-PLAY-000860818) at -823, and that they should strike a deal where, as Mr. Rosenberg put it, "the teams were not out competing with each other." Ex. 51 (GOOG-PLAY-007384816) at -818; *see also* Ex. 52 (Rosenberg Dep. Tr.) at 117:19–118:18. Samsung, in turn, understood Google's message: "So you're basically asking us to get out of the store business." Ex. 51 (GOOG-PLAY-007384816) at -818. Samsung is not the only significant market participant that Google has expressly urged not to compete with the Play Store. The architect of Google's early Android strategy wrote in 2009 that "this is why we ask our partners not to create their own app stores" even though

"[t]here are some antitrust issues" around it. Ex. 53 (GOOG-PLAY4-000810048). Google has continued to explicitly discourage other major industry players from competing with Play, including large would-be competitors such as [redacted] and [redacted]. *See, e.g.*, Ex. 54 (GOOG-PLAY-005566485) (Google messaging document noting that [redacted]; Ex. 55 (Bankhead Tr.) at 410:21-416:25 (discussing [redacted]

Seventh, Google leverages its power over app distribution into the aftermarket for services in connection with the sale of in-app content. Google's insistence that developers use its services for transacting in-app purchases in perpetuity reaps Google billions of dollars annually. Google's involvement in those sales is technologically unnecessary, but its requirement that all but a few developers use Google Play Billing to transact their in-app sales of digital content has to date been non-negotiable. And that requirement has allowed Google to impose itself as an unnecessary middleman, taking a supra-competitive toll of up to 30% for all in-app purchases in perpetuity, a fee that the Google Play team admitted has "no rationale other than copying Apple" and that "[redacted] Ex. 56 (GOOG-PLAY-000565541.R) at -552.R; Ex. 57 (GOOG-PLAY-004506631). Consumers, in turn, are stuck paying the lion's share of Google's overcharge.

IV. COMMON EVIDENCE OF CLASSWIDE ANTITRUST IMPACT

Plaintiffs rely on widely accepted economic models and other economic evidence common to the proposed classes to evaluate the effects of Google's conduct on consumers. Using common, economic evidence and widely accepted models, Dr. Hal Singer's analysis shows that in the absence of Google's anticompetitive conduct, Plaintiffs and class members would have paid lower prices and benefitted from enhancements to output, quality, and choice.

A. Common Evidence Establishes that Google's Take Rate Is Supra-Competitive

Consumer Plaintiffs will prove that Google's 30% headline take rate is supra-competitive through Dr. Singer's economic modeling of a "but-for world" free of Google's anticompetitive conduct in both the Android App Distribution Market and the In-App Aftermarket. For the Android

App Distribution Market, Dr. Singer uses the Rochet-Tirole model, a widely cited model that enables economists to estimate competitive prices in a two-sided market. *See Ohio v. American Express*, 138 S. Ct. 2274, 2281 (2018) (citing Jean-Charles Rochet and Jean Tirole, *Platform Competition in Two-Sided Markets*, 1 J. EUR. ECON. ASSN. 990, 1013 (2003)). For the In-App Aftermarket, Dr. Singer uses a standard single-sided economic model to determine competitive prices, because absent Google's anticompetitive insertion of itself into the transaction, aftermarket transactions would involve solely the developer and consumer. Ex. 2 (Singer Rpt.) at ¶ 33. In both cases, Dr. Singer's models generate competitive take rates that can determine classwide impact.

For the Android App Distribution Market, Dr. Singer calibrates the Rochet-Tirole model for the specific circumstances presented in this case and computes a competitive price in a competitive market with at least one other comparable app store. Ex. 2 (Singer Rpt.) at ¶¶ 184-194. Dr. Singer's analysis confirms that Google's dominance in the Android App Distribution Market enables Google "to extract a supra-competitive take rate on all paid App downloads." *Id.* at 72 ¶ 169. He determines that Google's average take rate on app sales would fall from [redacted]—which is slightly below the "headline" take rate of 30% due to discounts Google gives certain developers—to a "but-for take rate" of 23.4% for initial downloads of an app. *Id.* at 91 ¶ 205. Recognizing that the take rate for some developers deviates from the headline rate, Dr. Singer determines that the competitive take rates for these developers can be computed by applying the same proportion of real-world discounts to the competitive anchor rate. Ex. 2 (Singer Rpt.) at ¶¶ 194, 259.

To supplement his modeling, Dr. Singer compares platform take rates for similar two-sided digital platforms without the anticompetitive restraints Google imposes. *Id.* at ¶¶ 195-205 & Table 4. That comparative analysis shows Google's average [redacted] take rate is significantly higher than in comparable digital platforms in competitive environments, which range from 18% to 25%, confirming Google's supra-competitive pricing across the class. *Id.* at ¶ 205.

As he does with the Android App Distribution Market, Dr. Singer uses a well-accepted economic model to determine whether Google charges supra-competitive take rates in the In-App Aftermarket and to quantify any resulting damages to consumers. The model, which considers Google's marginal costs, demand elasticities, and other factors, incorporates the common-sense

economic principle that as competition increases, Google's take rate would be pushed "closer to the marginal cost of providing any services associated with In-App Content," while allowing Google a premium above marginal cost to account for consumer brand loyalty to Google. *Id.* at ¶¶ 208-13. Calibrating that model for the facts of this case, Dr. Singer concludes that Google's take rate in the competitive In-App Aftermarket would be 14.8%, considerably lower than Google's actual average take rate of [redacted] for in-app purchases. *Id.* at ¶¶ 206-220 & Table 5.

Again, Dr. Singer reviews common evidence on the relevant competitive landscape to confirm his results. While payment processing alone is not the only service provided by Google in the aftermarket, it is "a key component of [those] services." *Id.* at ¶ 221. Examining the take rates charged in the highly competitive payment processing industry with capable participants such as PayPal, Stripe, and Amazon Pay, Dr. Singer finds "materially lower" take rates ranging from approximately 2.9-3.95%, with varying fixed fees. *Id.* at 101, ¶ 221 & Table 6. This corroborates that Google's competitive take rate for in-app purchases would be significantly lower than the [redacted] that Google charges today. *Id.* at 101 ¶ 221 & Table 6. Because these potential competitors are currently foreclosed by virtue of Google's tie from competing with Google in the In-App Aftermarket for the full range of services provided by Google Play Billing, allowing entry by these and other firms in the but-for world would have effects classwide. Ex. 2 (Singer Rpt.) at ¶¶ 138, 207.

B. Common Economic Evidence Shows that All or Virtually All Class Members Are Injured by Google's Conduct

Because the precise ways in which competition would develop in a but-for world are difficult to predict due to Google's conduct, Dr. Singer models two potential but-for worlds, each with common methods.[12] First, a rival app store would force Google to compete by offering developers a lower take rate. Dr. Singer models consumer impact from this outcome by calculating pass-through from developers of savings from the lower take rates. *Id.* § V.D. Second, a rival app

[12] Plaintiffs are not required to predict exactly how competition would manifest itself. *See*, *e.g.*, *In re Elec. Books Antitrust Litig*., No. 11-MD-2293 (DLC), 2014 WL 1282293, at *27 (S.D.N.Y. Mar. 28, 2014) (it was "unsurprising" that an economic expert "did not develop an opinion about ... important features of a but-for world" because, "[a]fter all, the but-for world does not exist"); *J. Truett Payne Co., Inc. v. Chrysler Motors Corp*., 451 U.S. 557, 566 (1981) ("The vagaries of the marketplace usually deny us sure knowledge of what plaintiff's situation would have been in the absence of the defendant's antitrust violation.").

store would force Google to compete by offering consumers incentives in the form of coupons or reward points. Dr. Singer separately models the impact of these increased consumer incentives. *Id.* § V.E. Both models lead to the conclusion that all or virtually all consumers would benefit in a world free of Google's anticompetitive conduct.

First, Dr. Singer's models show that all or almost all class members would have benefited from lower prices resulting from reduced take rates. Ex. 2 (Singer Rpt.) at ¶¶ 223-226. Dr. Singer applies a commonly used logit demand system to model demand curves and developer pass-on of the reduced take rates discussed above for each of the thirty-five categories of apps the Google Play Store uses. *Id.* at V.D.3. Dr. Singer's calculations show the pass-through rates ranging from 25.0% to 95.8%, depending on the application category. *Id*. at ¶ 239 & Table 8. As with his other econometric modeling, Dr. Singer confirms his analysis by referencing real-world evidence of pass-through from 33% to 105% by apps that can avoid Google's In-App Aftermarket restrictions or Apple's similar restraints. *Id.* at ¶¶ 241-244.

Second, Dr. Singer models an alternative scenario in which Google competes for consumers by increasing its Google Play Points loyalty program. Using the Rochet-Tirole model, he finds classwide impact with an average of $0.76 in consumer savings per transaction. *Id.* at § V.E. In this competitive but-for world, which does not depend upon developer pass-on, Google would award Play Points in proportion to all purchases made by Class members, resulting in uniformly lower effective prices, and therefore impact all or virtually all Class members. *Id.* at ¶ 256.

C. Aggregate Damages Are Calculated on a Classwide Basis

Having determined the extent of the overcharge for both app and in-app purchases, Dr. Singer applies those percentages to Google's respective sales of apps and in-app content to calculate aggregate damages nationwide and by state. *Id.* at ¶ 274, Table 11, & Appendix 6. Dr. Singer calculates the aggregate damages for all consumers nationwide from Google's overcharge in the Android App Distribution Market to be $19 million for the Class Period through the end of 2020. *Id.* In the In-App Aftermarket, where consumers spend much more, he calculates aggregate nationwide damages to be $4.71 billion. *Id.* While unnecessary at this stage, Dr. Singer goes a step

further to show how individual damage calculations can be performed with a common formulaic method. *Id.* at § VII & Table 13.

ARGUMENT

A party seeking class certification must prove that the proposed class "satisfies the four requirements of Rule 23(a): (1) numerosity; (2) commonality; (3) typicality; and (4) adequacy of representation." *Senne v. Kansas City Royals Baseball Corp.*, 934 F.3d 918, 927 (9th Cir. 2019) (cleaned up). The "proposed class must also meet the requirements of one or more of the 'three different types of classes' set forth in Rule 23(b)." *Id*. Here, Plaintiffs propose two classes, one seeking money damages under Rule 23(b)(3), and the other seeking injunctive relief under Rule 23(b)(2). A (b)(3) damages class is appropriate when two additional elements are met: predominance and superiority. Fed. R. Civ. P. 23(b)(3). "Rule 23(b)(2), on the other hand, requires only that the party opposing the class have acted or refused to act on grounds that apply generally to the class." *Senne*, 934 F.3d at 928 (cleaned up).

I. THE REQUIREMENTS OF RULE 23(a) ARE MET IN THIS CASE

A. Rule 23(a)(1)'s Numerosity Requirement Is Satisfied

Rule 23(a)(1) requires the class to be "so numerous that joinder of all members is impracticable." Here, there are tens of millions of class members, so numerosity is easily satisfied.

B. Rule 23(a)(2)'s Commonality Requirement Is Satisfied

Rule 23(a)(2)'s commonality requirement is met because numerous questions of law and fact related to Google's unlawful conduct and its uniform impact on the class will be the centerpiece of any trial in this action. Rule 23(a)(2) requires the proposed class claims to pose "questions of law or fact common to the class." In other words, the "claims must depend upon a common contention," that "must be of such a nature that it is capable of classwide resolution—which means that determination of its truth or falsity will resolve an issue that is central to the validity of each one of the claims in one stroke." *Wal-Mart Stores, Inc. v. Dukes*, 564 U.S. 338, 350 (2011).

"[A] common contention need not be one that *will be* answered, on the merits, in favor of the class. It only must be of such a nature that it is *capable of* classwide resolution." *Alcantar v. Hobart Serv.*, 800 F.3d 1047, 1053 (9th Cir. 2015) (cleaned up) (emphasis added). "Where the

circumstances of each particular class member vary but retain a common core of factual or legal issues with the rest of the class, commonality exists." *B.K., by her next friend Tinsley v. Snyder*, 922 F.3d 957, 968 (9th Cir. 2019) (cleaned up). "For purposes of Rule 23(a)(2), even a single common question will do." *Wal-Mart*, 564 U.S. at 359 (cleaned up). "Antitrust liability alone constitutes a common question that 'will resolve an issue that is central to the validity' of each class member's claim 'in one stroke.'" *In re High-Tech Employee Antitrust Litig.*, 985 F. Supp. 2d 1167, 1180 (N.D. Cal. 2013) (quoting *Wal-Mart*, 564 U.S. at 350).

Plaintiffs' claims here depend upon several central, common questions of fact and law, discussed in more detail below, *infra* Part II.B, each of which focuses on Google's conduct and are capable of classwide resolution: (1) whether Google has monopoly power in the Android App Distribution Market and the In-App Aftermarket; (2) whether Google's contractual restrictions on manufacturers, carriers, and developers are unreasonable restraints of trade; and (3) whether Google's practices impacted class members and resulted in supra-competitive prices paid by consumers. Commonality is readily satisfied.

C. Rule 23(a)(3)'s Typicality Requirement Is Satisfied

Rule 23(a)(3) requires the claims of a class representative to be "typical" of the claims of the class. "Representative claims are 'typical' if they are reasonably coextensive with those of absent class members; they need not be substantially identical." *Ruiz Torres v. Mercer Canyons Inc.*, 835 F.3d 1125, 1141 (9th Cir. 2016) (cleaned up). "The test of typicality is whether other members have the same or similar injury, whether the action is based on conduct which is not unique to the named plaintiffs, and whether other class members have been injured by the same course of conduct." *Tinsley*, 922 F.3d at 970. Plaintiffs' Sherman Act and state claims are the same as other proposed class members. They are based on the same conduct by Google detailed above resulting in the same measures of damages to all and the identical need for injunctive relief against Google's anticompetitive conduct. As proposed in Dr. Singer's report, all class members will be utilizing a common method of demonstrating impact, further showing that the typicality standard is met in this case.

D. The Rule 23(a)(4) and 23(g) Adequacy Requirements Are Satisfied

The fourth Rule 23(a) requirements is that the representative plaintiffs "fairly and adequately protect the interests of the class." Rule 23(a)(4). Rule 23(g)(4) also requires class counsel to "fairly and adequately represent the interests of the class." "Resolution of two questions determines legal adequacy: (1) do the named plaintiffs and their counsel have any conflicts of interest with other class members and (2) will the named plaintiffs and their counsel prosecute the action vigorously on behalf of the class?" *In re TFT-LCD (Flat Panel) Antitrust Litig.*, 267 F.R.D. 291, 308 (N.D. Cal. 2010), *abrogated by In re ATM Fee Antitrust Litig.*, 686 F.3d 741, 755 n.7 (9th Cir. 2012) (cleaned up; quoting *Hanlon v. Chrysler Corp*., 150 F.3d 1011, 1020 (9th Cir. 1998)).

There are no conflicts of interest between the Plaintiffs and the proposed classes. And each of the proposed Class Representatives have, and will continue to, adequately represent the classes by taking an active part in the action. Each has actively participated in discovery, has familiarity with the nature of the action, reviewed the operative complaint before it was filed, and have made Google Play app or in-app purchases on Android devices.[13]

After having been appointed Co-Lead Interim Class Counsel under Rule 23(g)(3),[14] proposed Co-Lead Class Counsel have vigorously prosecuted the class claims since the inception of this action. They have engaged in nearly 22 months of extensive discovery coordination with Google and the other MDL plaintiffs, including: 17 depositions of Google fact witnesses; preparing for the depositions of numerous critical third-party witnesses; issuing more than 60 third-party subpoenas; meeting and conferring with Google and third parties on discovery issues multiple times a week; and reviewing approximately three million documents produced by Google and hundreds of thousands of documents produced by third parties. They have also hired experts who have worked over the course of a year and a half to develop opinions, including on antitrust injury

[13] *See, e.g.*, Ex. 58 (Carr Tr.) at 20:13-21:9; 23:7-11; 24:10-25:3; 31:4-5; 32:18-34:1; 122:25-123:3; Ex. 59 (Egerter Tr.) at 48:6-49:5; 52:15-53:7; 53:21-54:24; 126:5-126:16; Ex. 60 (Palmer Tr.) at 18:12-19:19:24; 25:3-27:4; 39:1-11; 49:17-22; 55:2-56:5; 193:16-21; Ex. 61 (Atkinson Tr.) 54:10-57:25; 69:1-70:21; 77:8-80:25; 97:4-99:3; 141:2-142:13; 191:6-24; Ex. 62 (Moglia Tr.) at 36:5-37:11; 40:19-46:11; 52:12-61:10; 82:1-83:8; 147:13-151:20; Ex. 63 (Iwamoto Tr.) 14:17-15:3; 37:11-38:20; 73:14-75:24; 89:4-20; 121:8-122:18; 131:13-25; 278:7-25.

[14] *See* ECF No. 128. Resumes covering their extensive experience in antitrust litigation are submitted again herewith Ex. 64 (Giulianelli Resume); Ex. 65 (Nam and Kaplan Fox Resume).

and damages and technical issues. Giulianelli Decl. ¶ 2. Finally, Counsel coordinated extensively with the Court Appointed Consumer Steering Committee,[15] and numerous State Attorneys' General to determine that it was in the best interests of consumers nationwide for Interim Class Counsel to enter into the Joint Prosecution Agreement, combining resources such that all parties could more effectively focus on their claims against Google.

The adequacy requirement has been satisfied, and Co-Lead Interim Class Counsel should be appointed Co-Lead Class Counsel.

II. RULE 23(b)(3)'S REQUIREMENTS ARE SATISFIED IN THIS CASE

A. The Predominance Requirement Is Met

The predominance requirement is met in this case because the vast majority of important questions will be resolved through classwide evidence. "The predominance inquiry asks whether the common, aggregation-enabling, issues in the case are more prevalent or important than the non-common, aggregation-defeating, individual issues." *Tyson Foods*, 577 U.S. at 453 (cleaned up).

> Predominance is not a matter of nose-counting. Rather, more important questions apt to drive the resolution of the litigation are given more weight in the predominance analysis over individualized questions which are of considerably less significance to the claims of the class.

In re Hyundai & Kia Fuel Econ. Litig., 926 F.3d 539, 557 (9th Cir. 2019) (cleaned up); *see also Messner v. Northshore Univ. Health Sys.*, 669 F.3d 802, 815 (7th Cir. 2012) ("The text of Rule 23(b)(3) itself contemplates that such individual questions will be present."). Courts cannot "decline to certify a class that will require determination of some individualized questions at trial, so long as such questions do not predominate over the common questions." *Olean*, 31 F.4th at 668.

The Supreme Court has said that "[p]redominance is a test readily met in certain cases alleging … violations of the antitrust laws." *Amchem Prods., Inc. v. Windsor*, 521 U.S. 591, 625 (1997). This is because the focus of an antitrust trial is the defendant's conduct. The same will be true in this case. Whether Google's conduct violated federal or state law are not questions that vary from class member to class member.

[15] The Court appointed Elizabeth Pritzker as liaison counsel, and Nanci Nishimura, Peggy Wedgworth, and George Zelcs as members of the steering committee. ECF No. 128.

1. Plaintiffs' Sherman Act Claims Present Common Questions That Will Predominate

Plaintiffs' Sherman Act Claims depend on proof of Google's anticompetitive conduct. "Section 2 monopolization claims readily lend themselves to common evidence" because they require proof of "(1) the possession of monopoly power in the relevant market and (2) the willful acquisition or maintenance of that power as distinguished from growth or development as a consequence of a superior product, business acumen, or historic accident." *In re Glumetza Antitrust Litig.*, 336 F.R.D. 468, 475 (N.D. Cal. 2020) (quoting *Alaska Airlines v. United Airlines*, 948 F.2d 536, 541 (9th Cir. 1991)). "Courts have routinely certified classes alleging that a monopolist's conduct caused a direct and uniform overcharge, as overcharge cases often rely on facts common across the class such as the existence of a relevant market, the defendant's monopoly power, and the existence of the illegal course of conduct." 6 W. RUBENSTEIN, *Newberg on Class Actions* § 20:28. (5th ed.) (collecting cases). Plaintiffs' Section 1 claims likewise turn on common questions. "[U]nder Section 1, a plaintiff must demonstrate: (1) that there was a contract, combination, or conspiracy; (2) that the agreement unreasonably restrained trade under either a per se rule of illegality or a rule of reason analysis; and (3) that the restraint affected interstate commerce." *Tanaka v. Univ. of S. Cal.*, 252 F.3d 1059, 1062 (9th Cir. 2001) (cleaned up).

The common questions identified above concerning the contours of the relevant markets, whether Google has power in those markets, and whether Google acquired and maintained power through a web of exclusionary contracts and conduct will be the overriding focus of any trial. "The violation turns on *defendants'* conduct and intent along with the effect on the market, *not* on individual class members." *In re Glumetza*, 336 F.R.D. at 475.

First, the definitions of the relevant markets—and Google's power in those markets—will be determined by common evidence. *See In re Apple iPhone Antitrust Litig.*, No. 4:11-cv-06714-YGR, 2022 WL 1284104, at *11 (N.D. Cal. Mar. 29, 2022) (expert's "opinion on market definition constitutes common proof that could resolve whether a relevant market exists"). Though Google does not challenge market definition for purposes of class certification, it may be disputed at trial, and resolution of that issue will turn on mountains of common evidence, as will a determination of Google's power in that market. Both will include common economic analysis of issues such as

Google's market share and ability to exclude competition and to raise prices above competitive levels. *Supra* Statement of Common Facts § II; *see In re Live Concert Antitrust Litig.*, 247 F.R.D. 98, 131 (C.D. Cal. 2007) ("the Court's focus is the process of defining the relevant market, and this process will clearly be predominated by common questions of law and fact").

Second, and central to the issues of this litigation, Google's pattern of unlawful conduct, including the seven major areas of conduct set out in Part III of the Statement of Common Facts above, will be the core of any trial in this action. The facts of Google's conduct with respect to carriers, OEMs, developers, and consumers, the application of the law to those facts, and determinations of whether Google's conduct violates the Sherman Act will be hotly disputed at trial but are all paradigmatic common questions. Each is "of such a nature that it is capable of class wide resolution—which means that determination of its truth or falsity will resolve an issue that is central to the validity of each one of the claims in one stroke." *Wal-Mart*, 564 U.S. at 350. Their determination will be based on common evidence of Google's conduct, evidence that will not vary from class member to class member.

Third, the antitrust impact of Google's conduct will be proven by common evidence. "A private antitrust plaintiff must prove the existence of antitrust injury, which is to say injury of the type the antitrust laws were intended to prevent and that flows from that which makes defendants' acts unlawful." *Ellis v. Salt River Project Agric. Improvement & Power Dist.*, 24 F.4th 1262, 1273 (9th Cir. 2022) (cleaned up). At the class certification stage, "[w]hat really matters is whether the class can point to common proof that will establish antitrust injury … on a class wide basis." *In re Capacitors III*, No. 17-md-02801-JD, 2018 WL 5980139, at *8 (N.D. Cal. 2018) (cleaned up). Through the expert testimony of Dr. Singer described above, *supra* § IV.A-B, Plaintiffs will present extensive statistical and econometric evidence that all or nearly all class members were injured. *Giuliano v. Sandisk Corp.*, No. C 10-02787 SBA, 2015 WL 10890654, at *17 (N.D. Cal. May 14, 2015) ("the relevant question is whether Plaintiffs have advanced a plausible methodology to demonstrate that antitrust injury and damages can be proven on a class-wide basis through the use of a common methodology"). As discussed above, Google documents and testimony, along with industry benchmarks, will further substantiate that Google caused antitrust injury. Consumers

"forced to pay an alleged monopolistic overcharge … have satisfied the antitrust injury requirement." *Goldwasser v. Ameritech Corp.*, 222 F.3d 390, 398 (7th Cir. 2000) (cleaned up).

***Fourth*,** damages also present common questions that can be resolved classwide through economic evidence. As discussed above, Dr. Singer has calculated aggregate overcharges using well-accepted economic methodologies that are common to the entire class. Ex. 2 (Singer Rpt.) at 130. He has also gone a step further and has shown how individualized damage calculations can be performed in a common, formulaic manner. *Id.* at 131-34, § VII & Table 13.

In short, each of the major questions of fact and law that will be resolved at trial will turn on common questions, which will predominate over any individual issues.

2. Plaintiffs' State Law Claims Present Common Questions That Will Predominate

Plaintiffs' state law claims present common questions because all such claims are governed by California law and implicate the same common facts discussed above. For the entire class period, Google's Terms of Services have provided that "California law will govern all disputes arising out of or relating to these terms, service-specific additional terms, or any related services, regardless of conflict of laws rules." *See* Ex. 66, Google Policies, *Terms of Services* (eff. Jan. 5, 2022), https://policies.google.com/terms; Ex. 67, Google Policies, *Google's Archived Terms of Services*, https://policies.google.com/terms/archive (links to March 31, 2020 Terms; October 25, 2017 Terms; and April 14, 2014 Terms). California law thus applies to all class members.

Plaintiffs' Cartwright Act claims (Counts 7-10) and federal claims are based on the same unlawful conduct. They therefore pose the same common predominating questions of fact identified above. *See*, *e.g.*, *Flagship Theatres of Palm Desert, LLC v. Century Theatres, Inc.*, 55 Cal. App. 5th 381, 399 (2020) (noting similarity of Sherman Act and Cartwright Act analysis of key antitrust issues). Plaintiffs' California Unfair Competition Law claim (Count 11) also presents common questions, including whether Google has engaged in deceptive or unfair trade practices.

3. Common Questions Predominate on Impact and Damages

As described in Part IV.B., Dr. Singer has established a common economic framework to demonstrate that all or nearly all proposed class members are impacted by Google's restrictions.

And as described in Part IV.C., he has also shown the aggregate damages suffered by proposed class members, as well as a methodology to calculate individual damages.

Consumers expect that Google will use the report of its economist, Dr. Burtis, to argue that there is too much heterogeneity in the market to find common impact in the counterfactual but-for world. But "variations in products and complexities in a distribution chain do not preclude an estimation of whether an overcharge impacted end purchasers." *In re Disposable Contact Lens Antitrust Litig.*, 329 F.R.D. 336, 420 (M.D. Fla. 2018) (quoting *In re TFT-LCD (Flat Panel) Antitrust Litig.*, 267 F.R.D. 583, 603 (N.D. Cal. 2010), amended in part, No. M 07-1827 SI, 2011 WL 3268649 (N.D. Cal. July 28, 2011)); *see also In re Suboxone (Buprenorphine Hydrochloride and Nalaxone) Antitrust Litig.*, MDL No. 2445, 13-md-2445, 2019 WL 4735520, at *31 (E.D Pa. Sep. 27, 2019) (noting that "district courts have ... rejected contentions ... that differences in purchasing decisions defeated class certification on issues of antitrust impact" and citing cases). None of the hypothetical issues Google may raise defeat predominance in this case.

First, Dr. Burtis claims that Google would engage in highly individualized take-rate negotiations with developers in a competitive but-for world, complicating common impact. Ex. 7 (Burtis Rpt.) § VI.A. But Dr. Burtis's speculation does not track Google's behavior in the actual world—where it has kept its take rate largely uniform, with adjustments largely in the context of formulaic programs or for a very select group of large developers. Ex. 3, Singer Reply Rpt. ¶¶ 8-10. Nor does she explain why Google would change that approach when faced with competition or how it would be feasible for Google to individually negotiate with thousands of developers. In any event, Dr. Singer's methodology allows for common calculation of reduced take rates from the "headline take rate" in the but-for world. As Dr. Singer explains, "it is reasonable to assume that those developers that secured discounts off the inflated rate of 30 percent would have also secured the discounts off a lower, headline rate." Ex. 3, Singer Reply Rpt. ¶ 7.

Consistent with this, the Ninth Circuit has held that antitrust schemes can affect "all market participants, creating an inference of class-wide impact even when prices are individually negotiated," and it rejected "different bargaining power" among market participants as a rationale for denying class certification in antitrust cases. *Olean*, 31 F.4th at 671, 677 (citing *In re Urethane*

Antitrust Litig., 768 F.3d 1245, 1254 (10^{th} Cir. 2014); *In re Fine Paper Antitrust Litig.*, 82 F.R.D. 143, 151-52 (E.D. Pa. 1979) (granting certification despite concerns about diversity of product, marketing practices, and pricing in conspiracy cases because conspiracy is "the overriding predominant question"). The Ninth Circuit further held that there are well-accepted econometric techniques—some of which Dr. Singer utilized in this case—for demonstrating "antitrust impact in markets with individualized differences among purchasers." *Olean*, 31 F.4^{th} at 674; *see also In re Vitamins Antitrust Litig.*, 209 F.R.D. 251, 260 (D.D.C. 2002) (rejecting argument that a putative class of "diverse businesses, facing different competitive environments and paying widely different prices" defeats certification).

***Second*,** Dr. Burtis argues that consumers who exclusively make low-dollar purchases would be unharmed because of allegedly high per-transaction fees for low-dollar-value transactions from third-party payment processors. But this, too, ignores prevailing practices involving "micropayment" rates. Ex. 3 (Singer Reply) at ¶ 15. Further, in a more competitive world, economics predicts that competing payment processors would eschew minimum per transaction fees (as they already have) if they want to win the business of lower-priced apps. Ex 3, Singer Reply Rpt. ¶ 14-16. Differences in payment processing costs for small transactions—to the extent they even exist—do not require individualized analysis. *See In re Mercedes-Benz Antitrust Litig.*, 213 F.R.D. 180, 187 (D. N.J. 2003) (rejecting contention that "the idiosyncrasies of each transaction also preclude the possibility that common issues will predominate on the element of impact.").

***Third*,** Dr. Burtis argues that individualized inquiry is required to determine some developers' marginal costs, pricing strategies, and competitive conditions, and, accordingly, the degree of pass-on to consumers. Ex. 5, Burtis Rpt. §§ VI.B.1, VI.B.2, VI.B.3. But as explained in Dr. Singer's reply report, many of Dr. Burtis's assertions rest on shaky or speculative factual foundations, and Dr. Singer's methodology is not compromised by any such differences. Ex. 3, Singer Reply Rpt. ¶¶ 21-32. For instance, while Dr. Burtis claims that developer "focal point" pricing ending in \$.99 would prevent pass-on for many applications in a competitive world, she ignores the evidence that (1) Google until very recently prohibited U.S. developers from charging less than \$.99; and (2) even in the *actual* world, when developers are allowed to steer, a significant number

of them do divert from this pricing. *Id.* ¶¶ 26-29. If developers could save money by steering to lower-cost alternatives, which Google currently forbids, they would do so. Her analysis is also contrary to economic logic in the literature on "psychological pricing." *Id.* ¶ 30.[16]

As a variation on this theme, Dr. Burtis criticizes Dr. Singer's models' reliance on average inputs and results for his but-for take rates. Ex. 5, Burtis Rpt. ¶ 340. Her similar arguments have been rejected in *D&M Farms v. Birdsong Corp.*, Civil Action No. 2:19-cv-463, 2020 WL 7074140, at *1, *6, *8 (E.D. Va. Dec. 2, 2020) (granting class certification and disregarding Dr. Burtis's accusation that plaintiffs relied upon averages attributable to the entire class instead of those applicable to each proposed class member); and *In re Static Random Access Memory (SRAM) Antitrust Litig.*, 264 F.R.D. 603, 614, 616, 617 (N.D. Cal. 2009) (granting class certification while rejecting Dr. Burtis's argument that using aggregated and averaged data in an econometric model could yield "false-positive pass-through").

Finally, Dr. Burtis claims that actions Google might take in a but-for competitive world, such as [redacted] would differentially affect consumers, leaving some uninjured. Ex. 7, Burtis Rpt. §§ VI.C.4, VI.C.5, VI.D. Dr. Burtis's parade of horribles is based entirely on speculation and runs counter to standard economic principles. As Dr. Singer notes, "Economic principles prescribe that as competition increases, firms compete more vigorously for customers on all dimensions, including services." Ex. 3, Singer Reply Rpt. ¶ 64. It is, therefore, unlikely that Google would take steps [redacted] in the face of competition, when it hasn't done so in the actual world. Ex. 3, Singer Reply Rpt. ¶¶ 46-49, 64-66. Not only has Google declined to take these steps, in some cases [redacted] Ex. 3, Singer Reply Rpt. ¶¶ 41-44, 52 & n.97. Dr. Burtis's

[16] Although the Court in *In re Apple Iphone Antitrust Litig.*, 2022 WL 1284104, at *8, faulted plaintiffs for their expert's failure to consider focal-point pricing in his model, there the plaintiffs' expert testified that he would expect to see focal-point pricing in a but-for world, and the Court found that the record had "overwhelming evidence suggest[ing] that developers would choose to price their apps at focal points ending in 99 cents." Neither is true here. Plaintiffs have established a record demonstrating that focal-point pricing is not integral to developers' pricing in a but-for world where Google's anticompetitive conduct no longer constrains developers' actions, and, therefore, it is unnecessary to include in a pricing model. Ex. 3 (Singer Reply) ¶¶ 26-30.

speculation that Google *might* take steps in the but-for world that differentially impact the class, when it has rejected those steps in the actual world, does not defeat the predominance of common questions. *In re Microcrystalline Antitrust Litig.*, 218 F.R.D. 79, 92-93 (E.D. Pa. 2003) ("Antitrust cases nearly always require some speculation as to what would have happened under competitive conditions … but this is not fatal to class certification.")

Any attempt by Google to inject individualized issues is unavailing. The central issues in this case—Google's anticompetitive monopolization and its common impact on consumers—will be tried through evidence common to the class, which is why monopolization cases are readily certified. *See In re Glumetza*, 336 F.R.D. at 475 (N.D. Cal. 2020).

B. The Superiority Requirement Is Met

Rule 23(b)(3) provides a non-exhaustive list of four factors "[a]s an aid for determining the superiority question." *Kamm v. Calif. City Dev. Co.*, 509 F.2d 205, 212 (9th Cir. 1975). "Rule 23(b)(3)'s superiority test requires the court to determine whether maintenance of this litigation as a class action is efficient and whether it is fair." *Wolin v. Jaguar Land Rover N. Am., LLC*, 617 F.3d 1168, 1175-76 (9th Cir. 2010). As for class members' interests in individually controlling the prosecution or defense of separate actions, "where recovery on an individual basis would be dwarfed by the cost of litigating on an individual basis, this factor weighs in favor of class certification." *Id.* at 1175. The second and third factors—litigation already begun and desirability of concentrating the litigation in the particular forum—favor certification. Google antitrust litigation is already "concentrated" before this Court. The first and fourth factors—class members' interests in individually controlling the prosecution and manageability—also favor certification. Litigating Plaintiffs' claims as a class action will be fair, efficient, and superior to other available methods for fairly and efficiently adjudicating the controversy.

III. THE REQUIREMENTS OF RULE 23(b)(2) ARE SATISFIED IN THIS CASE

A class seeking injunctive or declaratory relief should be certified if "the party opposing the class has acted or refused to act on grounds that apply generally to the class, so that final injunctive relief or corresponding declaratory relief is appropriate respecting the class as a whole." Rule 23(b)(2). "The key to the (b)(2) class is the indivisible nature of the injunctive or declaratory

remedy warranted—the notion that the conduct is such that it can be enjoined or declared unlawful only as to all of the class members or as to none of them." *Wal-Mart*, 564 U.S. at 360 (cleaned up). "These requirements are unquestionably satisfied when members of a putative class seek uniform injunctive or declaratory relief from policies or practices that are generally applicable to the class as a whole." *Parsons v. Ryan*, 754 F.3d 657, 688 (9th Cir. 2014).

In this case, Rule 23(b)(2)'s straightforward requirement is plainly met. The evidence Plaintiffs have presented in support of their certification motion, discussed more fully above, demonstrates that Google has acted "or refused to act on grounds generally applicable to the class, thereby making appropriate final injunctive relief or corresponding declaratory relief with respect to the class as a whole." Fed. R. Civ. P. 23(b)(2).

Google's conduct at issue is not specific to any consumer. Google has acted "on grounds that apply generally to the class." If that conduct violates the Sherman Act, the Cartwright Act, or Unfair Competition Law then a declaration that Google's conduct is unlawful, and an injunction enjoining Google from continuing to engage in that conduct, would be "appropriate respecting the class as a whole." *Parsons*, 754 F.3d at 689. "Because an injunction would offer all class members 'uniform relief' from [ongoing antitrust harms], class certification is appropriate under Rule 23(b)(2)." *In re NCAA Student-Athlete Name & Likeness Licensing Litig.*, No. C 09-1967 CW, 2013 WL 5979327, at *7 (N.D. Cal. Nov. 8, 2013) (certifying (b)(2) class in Sherman Act case).

CONCLUSION

This case presents numerous, pivotal common questions of law and fact. The evidence at trial will plainly focus on Google's conduct, not issues or evidence specific to individual plaintiffs. Moreover, as Dr. Singer has demonstrated, Plaintiffs will rely on economic models that generate common answers to those questions. Plaintiffs have, therefore, demonstrated that the injuries they and class members have sustained are classwide and warrant classwide relief. The requirements of Rule 23 are satisfied in this case, and the Court should therefore certify the proposed classes.

Respectfully submitted,

By: */s/ Karma M. Giulianelli*
BARTLIT BECK LLP
Karma M. Giulianelli (SBN 184175)
Glen E. Summers (SBN 176402)
Jameson R. Jones (pro hac vice)
1801 Wewatta St., Suite 1200
Denver, CO 80202
Telephone: (303) 592-3100
Facsimile: (303) 592-3140
karma.giulianelli@bartlitbeck.com
glen.summers@bartlitbeck.com
jameson.jones@bartlitbeck.com

BARTLIT BECK LLP
John Byars (pro hac vice)
Lee Mason (pro hac vice)
54 W. Hubbard St., Suite 300
Chicago, IL 60654
Telephone: (312) 494-4400
Facsimile: (312) 494-4440
john.byars@bartlitbeck.com
lee.mason@bartlitbeck.com

KOREIN TILLERY, LLC
George A. Zelcs (pro hac vice)
Randall Ewing, Jr. (pro hac vice)
205 North Michigan, Suite 1950
Chicago, IL 60601
Telephone: (312) 641-9750
Facsimile: (312) 641-9751
gzelcs@koreintillery.com
rewing@koreintillery.com

Stephen M. Tillery (pro hac vice)
Michael E. Klenov (SBN 277028)
Carol O'Keefe (pro hac vice)
505 North 7th Street, Suite 3600
St. Louis, MO 63101
Telephone: (314) 241-4844
Facsimile: (314) 241-3525
stillery@koreintillery.com
mklenov@koreintillery.com
cokeefe@koreintillery.com

By: */s/ Hae Sung Nam*
KAPLAN FOX & KILSHEIMER LLP
Hae Sung Nam (pro hac vice)
Robert N. Kaplan (pro hac vice)
Frederic S. Fox (pro hac vice)
Donald R. Hall (pro hac vice)
Aaron L. Schwartz (pro hac vice)
850 Third Avenue
New York, NY 10022
Tel.: (212) 687-1980
Fax: (212) 687-7715
hnam@kaplanfox.com
rkaplan@kaplanfox.com
ffox@kaplanfox.com
dhall@kaplanfox.com
aschwartz@kaplanfox.com

KAPLAN FOX & KILSHEIMER LLP
Laurence D. King (SBN 206423)
Kathleen A. Herkenhoff (SBN 168562)
1999 Harrison Street, Suite 1560
Oakland, CA 94612
Telephone: 415-772-4700
Facsimile: 415-772-4707
lking@kaplanfox.com
kherkenhoff@kaplanfox.com

PRITZKER LEVINE, LLP
Elizabeth C. Pritzker (SBN 146267)
Bethany Caracuzzo (SBN 190687)
Caroline Corbitt (SBN 305492)
1900 Powell Street, Suite 450
Emeryville, CA 94608
Telephone: (415) 805-8532
Facsimile: (415) 366-6110
ecp@pritzkerlevine.com
bc@pritzkerlevine.com
ccc@pritzkerlevine.com

COTCHETT, PITRE & MCCARTHY, LLP
Nanci E. Nishimura (SBN 152621)
Adam J. Zapala (SBN 245748)
Elizabeth T. Castillo (SBN 280502)
James G. Dallal (SBN 277826)
San Francisco Airport Office Center
840 Malcolm Road, Suite 200
Burlingame, CA 94010
Telephone: (650) 697-6000
Facsimile: (650) 697-0577
nnishimura@cpmlegal.com
azapala@cpmlegal.com
ecastillo@cpmlegal.com
jdallal@cpmlegal.com

MILBERG PHILLIPS GROSSMAN LLP
Peggy J. Wedgworth (pro hac vice)
Robert A. Wallner (pro hac vice)
Elizabeth McKenna (pro hac vice)
Blake Yagman (pro hac vice)
Michael Acciavatti (pro hac vice)
One Penn Plaza, Suite 1920
New York, New York 10119
Telephone: 212-594-5300
Facsimile: 212-868-1229
pwedgworth@milberg.com
rwallner@milberg.com
emckenna@milberg.com
byagman@milberg.com
macciavatti@milberg.com

Counsel for the Proposed Class

CERTIFICATE OF SERVICE

The undersigned certifies that a true and correct copy of the foregoing was served on May 26, 2022 upon all counsel of record via the Court's electronic notification system.

/s/ Karma M. Giulianelli

REDACTED VERSION

Exhibit A2
to
C. Cramer Declaration

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

IN RE GOOGLE PLAY STORE ANTITRUST LITIGATION THIS DOCUMENT RELATES TO: *In re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD	No. 3:21-md-02981-JD **CLASS CERTIFICATION REPORT OF HAL J. SINGER, PH.D.** Judge: Hon. James Donato

HIGHLY CONFIDENTIAL UNDER PROTECTIVE ORDER

INTRODUCTION

1. Google Play (the "Play Store"), owned and operated by Google,[1] is the largest distributor of Android-compatible software applications for mobile devices ("Apps") and the only store that can reach virtually every Android mobile user outside of China. With some exceptions, Google generally takes 30 percent of all revenues on the original sale and downloading of Apps from the Play Store and the sale of digital content within Apps ("In-App Content").[2] Counsel for Mary Carr, Daniel Egerter, Zack Palmer, Serina Moglia, Matthew Atkinson, and Alex Iwamoto, on behalf of themselves and all others similarly situated (the "Consumer Plaintiffs" or "Class"), have asked me to assess the competitive effects of the various restrictions Google enforces (collectively, the "Challenged Conduct"), to extract these "take rates,"[3] and, in particular, to assess whether, as a result of the Challenged Conduct, consumers have overpaid for the initial downloads of Apps through the Play Store and for purchases of In-App Content.

2. In this report, I rely on common economic evidence and methods for my preliminary conclusions herein. Common economic evidence and methods confirm that Google has monopoly power in the market for the sale and distribution of Apps for Android mobile devices (the "Android App Distribution Market") and that it has gained and maintained such power through the imposition of exclusionary contractual provisions and artificial technological barriers that unnecessarily impede the distribution of applications outside of the Play Store. Similarly common economic evidence and methods show that Google has extended its power in the Android App Distribution Market into the ancillary aftermarket for services in support of consummating purchases of In-App Content (the "In-App Aftermarket"). I determine that Google has gained and maintained significant market power in the In-App Aftermarket through anticompetitive, exclusionary contractual restrictions (the "Aftermarket Restrictions") that function as an economic tie-in of Google's Android App Distribution Market services to the In-App Aftermarket. In the event that the factfinder concludes that the Android App Distribution Market and In-App Aftermarket are not two separate markets, I present alternative models that can be applied to a single, combined market, again using common economic evidence and methods. I further conclude that, in the absence of Google's anticompetitive conduct, the Consumer Plaintiffs would have paid lower prices for both Apps and In-App Content, and would also have benefitted from enhancements to output, quality, and consumer choice. This conclusion holds regardless of whether there are two relevant markets or just one.

---

1. Google includes Google, LLC, Google Ireland Ltd., Google Commerce Ltd., Google Asia Pacific Pte. Ltd., and Google Payment Corp.

2. As shown in Tables 3 and 5, *infra*, Google's average take rate across all transactions on the Play Store during the Class Period (August 16, 2016 through December 31, 2020) exceeds 29 percent for both initial downloads and In-App Content. The reason why the average take rate is slightly below 30 percent is that Google gives discounts from its baseline rate under certain limited circumstances described in this report. While Google's online policies have changed over time, In-App Content as used in this report is consistent with what Google refers to as "in-app purchases." Google's current policy states: "Play-distributed apps requiring or accepting payment for access to in-app features or services, including any app functionality, digital content or goods (collectively "in-app purchases"), must use Google Play's billing system for those transactions" subject to discrete exceptions. *See* Play Console Help, *available at* https://support.google.com/googleplay/android-developer/answer/9858738.

3. To describe its price to developers, Google uses the terminology "revenue share," which reflects the fact that Google is taking a piece of the developers' revenues, injecting itself as a "partner" in the customer-developer relationship. Google also uses the term "revenue share" to refer to the portion of the take rate Google has shared with mobile carriers and original equipment manufacturers. For ease of exposition, I use the term "take rate" to capture Google's price charged to developers.

QUALIFICATIONS

3. I am a managing director at Econ One, an economic consulting firm, and an adjunct professor at the McDonough School of Business at Georgetown University, where I teach advanced pricing to MBA candidates. A copy of my CV, including a list of my expert testimony since 2012, is attached to this report as Appendix 1.

4. I am an applied microeconomist with an emphasis on industrial organization and regulation. In an academic capacity, I have published several books and book chapters spanning a range of industries and topics, and my articles have appeared in dozens of legal and economic journals. My competition-related articles have appeared in multiple American Bar Association (ABA) Antitrust Section journals, and I have been a panelist at several ABA Antitrust Section events. In a consulting capacity, the American Antitrust Institute (AAI) nominated me for antitrust practitioner of the year in 2013 in the economist category for my work in *Tennis Channel v. Comcast*, and the AAI named me as co-Honoree in the same category in 2018 for my work *In Re Lidoderm Antitrust Litigation*.

5. I have testified as an economic expert in state and federal courts, as well as before regulatory agencies. Federal courts have certified multiple classes in reliance on my economic testimony on antitrust impact and damages.[4] I also have testified many times before Congress on competition policy, most recently at a February 2021 hearing titled "Reviving Competition, Part 1: Proposals to Address Gatekeeper Power and Lower Barriers to Entry Online," held by the House Subcommittee on Antitrust; and at a February 2022 hearing titled "Breaking the News – Journalism, Competition, and the Effects of Market Power on a Free Press," held by the Senate Subcommittee on Competition Policy, Antitrust, and Consumer Rights. Finally, I have served as an antitrust expert to Competition Bureau Canada and to several state Attorneys General.

6. Econ One Research is being compensated for my work in this matter at my standard hourly rate of $885. My compensation does not depend on the outcome of this litigation.

7. The materials I relied upon in forming my opinions are noted in the footnotes throughout this report or otherwise listed in Appendix 2. I provide this report to assist in evaluating the suitability of the case for class certification. As discovery is ongoing, I reserve the right to supplement, expand, or amend my opinions.

BACKGROUND

8. Mobile devices, and smartphones in particular, have become ubiquitous in daily life, essential providers of communication, entertainment, and information. These hand-held

---

4. *See Meijer, Inc. v. Abbott Laboratories*, No. C 07-5985 CW, 2008 WL 4065839 (N.D. Cal. Aug. 27, 2008) (order granting plaintiffs' motion for class certification); *Natchitoches Parish Hosp. Serv. Dist. v. Tyco Intl., Ltd.*, 262 F.R.D. 58 (D. Mass. 2008) (granting motion to certify class); *In re Delta/AirTran Baggage Fee Antitrust Litig.,* 317 F.R.D. 665 (N.D. Ga. 2016) (same); *Johnson v. Arizona Hosp. and Healthcare Assoc.* No. CV 07-1292-PHX-SRB, 2009 WL 5031334 (D. Ariz. July 14, 2009) (granting in part motion for class certification); *Southeast Missouri Hospital and St. Francis Medical Center v. C.R. Bard*, No. 1:07cv0031 TCM, 2008 WL 4372741 (E.D. Mo. Sept. 22, 2008) (granting in part and denying in part motion for class certification); and *In re Lidoderm Antitrust Litig.*, No. 12-md-02521, 2017 WL 679367 (N.D. Cal. Feb. 21, 2017) (order granting motions for class certifications and denying Daubert motions); *Cung Le et al. v. Zuffa, LLC d/b/a Ultimate Fighting Championship,* Minute Entry, 2:15-cv-01045-RFB-BNW (D. Nev. Dec. 10, 2020), ECF No. 781 (announcing the court's intention to grant the plaintiffs' motion for class certification). As of the time of this report, the court has not issued the written opinion.

portable computers can be used almost anywhere, allowing us to connect to the internet from any location that offers the requisite cellular or wi-fi network. Whether you want to buy stock in GameStop, check the score of the San Francisco 49ers game, or call your mother—mobile devices do it all.

9. Like the mini-computers they are, mobile devices are comprised of both hardware and software. The hardware typically consists of an LCD or OLED flat screen and some combination of physical buttons, digital keypads, and, ever more frequently, a touchscreen interface.[5] As with computers, mobile devices are controlled by an operating system ("OS"), software that manages the device's hardware and software resources. But the real work—and play—on mobile devices is performed by consumer "applications" or more commonly "apps," software programs designed to perform a myriad of functions. Apps are typically displayed on the device screen through a representative image, known as an icon, and can be called by the user through a touch, a tap, or the click of a button.

10. Mobile device operating systems dictate how an app must be developed to function on the device. However, operating systems are not cross-compatible; an app designed to run on one operating system must be substantially re-written to function on a separate, distinct operating system.[6] Prior to the launch of Android mobile devices, individual cellular carriers and original equipment manufacturers ("OEMs") controlled the industry, providing not only the hardware for mobile devices, but also writing most of the software for their own devices.

11. Today, the only two major mobile device operating systems globally (excluding China) are Apple's iOS and Android.[7] Apple is the exclusive hardware producer of its popular iPhone (smartphone) and iPad (tablet) devices, which are powered by Apple's proprietary iOS operating system. Unlike iOS, the Android operating system is allegedly "open source," meaning that anyone can inspect, modify, or enhance the source code to manipulate the software. While an open-source version of Android exists, the vast majority of OEMs manufacture Android OS devices that meet Google's compatibility requirements and are preloaded with a proprietary suite of apps and interfaces that Google designed specifically for mobile devices ("Google Mobile

---

5. *See* IGI Global, *What is Mobile Devices*, *available at* www.igi-global.com/dictionary/use-of-apps-and-devices-for-fostering-mobile-learning-of-literacy-practices/18837 ("A mobile device is a computing device small enough to hold and operate in the hand. Typically, any handheld computer device will have an LCD or OLED flat screen interface, providing a touchscreen interface with digital buttons and keyboard or physical buttons along with a physical keyboard.").

6. *See* Lionel Valdellon, *What Are the Different Types of Mobile Apps? And How Do You Choose?*, CLEVERTAP, (Nov. 1, 2020), *available at* clevertap.com/blog/types-of-mobile-apps/("Native apps are built specifically for a mobile device's operating system (OS). […] [T]he problem with native apps lies in the fact that if you start developing them, you have to duplicate efforts for each of the different platforms. The code you create for one platform cannot be reused on another. This drives up costs. Not to mention the effort needed to maintain and update the codebase for each version."). *See also* Jason Turnquist, *How Much Does App Development Cost?*, FYRESITE, (July 31, 2020), *available at* www.fyresite.com/how-much-does-app-development-cost/ (Detailing how Uber might seem like one app, but is in reality four apps: "a native iOS app for drivers, a native Android app for drivers, a native iOS app for riders, and a native android app for riders," where each new permission [app] created increases the price to develop it.).

7. *See, e.g.,* Sherisse Pham, *Google now has two apps in China, but search remains off limits*, CNN BUSINESS, (May 31, 2018), *available at* money.cnn.com/2018/05/31/technology/google-in-china-files-app/index.html ("The company's own app store, Google Play, remains blocked in China[.]").

Services" or "GMS").[8] For purposes of this report, I refer to devices that are pre-loaded with the GMS suite of apps as ("Google Android"); these devices make up over 70 percent of all mobile devices in the global market.[9] Collectively, Google's Android and Apple's iOS make up 99 percent of mobile devices worldwide.[10]

12. The functionality and user enjoyment derived from a mobile device is highly dependent upon the range and quality of apps available on it. In addition to producing a mobile operating system, Google has created a distribution channel for delivery of Android-compatible apps developed by third parties, and developed its own universe of Apps, for Google Android. Google itself has developed some of the most popular Android- and iOS-compatible apps, including Google Search, Google Maps, Chrome, YouTube, and Gmail.

13. When purchased by the consumer, mobile devices come pre-loaded with a variety of apps pre-positioned on the device's "home screens," each of which is accessed by swiping your thumb. The first screen is known as the "default home screen." Because pre-loaded apps—whether on the home screen or otherwise—are automatically available to all the users who purchase the device, app developers would find it advantageous to have their apps pre-installed by the OEMs (though, as detailed below, preinstallation is not an option for the majority of developers).[11] Similarly, the placement of an app in a prominent place on a device's home screen makes it more likely that consumers will open and engage with the app, meaning the initial icon placement can strongly influence an app's overall usage and popularity.[12] Pre-installation and default placement of pre-loaded apps are significant factors in determining an app's adoption by consumers and the app's ultimate success.

14. In addition to proprietary Apple and Google apps, independent software developers create and code a broad universe of apps for both operating systems. However, the Android and Apple operating systems are not compatible, meaning that software developers must create independent versions of their apps to operate on each system. For the vast majority of developers, expending the time and resources necessary to create an app for a particular operating system depends upon the number of consumers using a device running that operating system. Given the reach of Google Android and Apple iOS devices, most large developers currently create and provide apps for both systems.

---

8. Android, *The best of Google, right on your devices*, *available at* www.android.com/gms/ ("Google Mobile Services (GMS) is a collection of Google applications and APIs that help support functionality across devices.").

9. Statista, *Mobile operating systems' market share worldwide from January 2012 to June 2021, available at* www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/ [hereafter *Statista Mobile OS Shares*]. These data include China. For the purposes of the relevant markets here, however, I exclude China because, by government decree, Apps downloaded there cannot be supported by the Play Store.

10. *Id*. It bears noting that Android and Apple jointly dominate the marketplace for mobile operating systems in China as well. *See* Statista, *Market share of mobile operating systems in China from January 2013 to March 2021*, *available at* www.statista.com/statistics/262176/market-share-held-by-mobile-operating-systems-in-china/ (showing Android and Apple accounting for over two thirds and one fifth of the Chinese mobile operating market, respectively).

11. *See, e.g.,* GOOG-PLAY-010801568 at GOOG-PLAY-010801570 (Facebook presentation noting that "we invest in Facebook Preloads to ensure we can provide the best experience of our apps on Android to people globally (including those without a Play account)."); GOOG-PLAY-001404176 ("Preloading remains valuable to users, and hence OEMs, despite full unbundling because most users just use what comes on the device. People rarely change defaults.").

12. *See, e.g.,* GOOG-PLAY-006355073 ("Fortunately, we'll always have the placement / pre-install advantage which is 90% of the battle.").

15. At the level of mobile devices running an operating system, Google Android devices, manufactured by OEMs such as Samsung, and Apple devices compete with one another for users.[13] However, once a consumer has elected to purchase a mobile device using either Google Android or Apple iOS, they are effectively locked-in to that ecosystem for that device. Because the two ecosystems are incompatible, paid apps cannot be transferred by a user from one system to another.[14] The user interfaces for an Apple iOS device and a Google Android device function differently, requiring significant expenditures of time and effort by users to learn and master. Users also invest money and time in identifying, buying and/or installing apps and content, including games, music, and videos, that may have to be repurchased or reinstalled upon switching operating systems. As a result of these factors, economic analysis and application of standard economic tests for market definition show that the market for Android apps and for the distribution of those apps is distinct and separate from the market for app distribution on Apple's proprietary iOS.

16. Google is the pioneering creator of the Google search engine. Google's business strategy has focused on providing free software and digital services,[15] which in turn generated information about users, ranging from personal identifying information like email and billing addresses, to areas of interest, to discrete preferences in products and services. Google does not sell its raw user information,[16] choosing instead to monetize this information via its vast advertising business. Google provides advertising inventory including space on its Google Search result pages and its YouTube video platform. It also provides tools used by both advertisers and online publishers in the purchase and sale of digital display advertising. Google's vast trove of user information, gathered across its range of free software products, including the Play Store, enhances the value of all Google advertising products.[17]

---

13. The competition does not extend to all users. Most Android phones are at a price point below the cheapest iPhone. And most Android users outside the U.S. could not afford an iPhone even if they wanted it. Given its wealth, the United States is an outlier in this respect.

14. Apple cannot guarantee even that all free apps will be ported. *See* Apple, *Move from Android to iPhone, iPad or iPod touch*, *available at* support.apple.com/en-gb/HT201196 ("Here's what gets transferred: contacts, message history, camera photos and videos, photo albums, files and folders, accessibility settings, display settings, web bookmarks, email accounts and calendars. If they're available on both Google Play and the App Store, *some of your free apps* will also be transferred. After the transfer is complete, you can download any free apps that were matched from the App Store.") (emphasis added). Likewise, Google's own Pixel switching service cannot transfer paid apps from iOS. *See* GOOG-PLAY-004147888 at GOOG-PLAY-004147897; Pixel Phone Help, *Transfer Data from an iPhone to a Pixel, available at* support.google.com/pixelphone/answer/7129740#what_doesnt_copy&zippy=%2Cwhat-wont-copy-during-setup ("What won't copy during setup" includes "Paid apps" and "Unpaid apps not matched on the Play Store").

15. 2004 Founder's IPO Letter, From the S-1 Registration Statement, *available at* abc.xyz/investor/founders-letters/2004-ipo-letter/.

16. Google Safety Center, *Ads That Respect Your Privacy*, *available at* safety.google/privacy/ads-and-data/ ("We never sell your personal information.").

17. *See, e.g.*, Megan Graham & Jennifer Elias, *How Google's $150 billion advertising business works*, CNBC, (May 18, 2021), *available at* www.cnbc.com/2021/05/18/how-does-google-make-money-advertising-business-breakdown-.html. *See also* Competition & Markets Authority, *Online platforms and digital advertising: Market study final report*, (July 1, 2020), 1-437, 228, *available at* assets.publishing.service.gov.uk/media/5fa557668fa8f5788db46efc/Final_report_Digital_ALT_TEXT.pdf ("Google has tags (including as a third-party) on over 80% of websites and over 85% of apps on the Play Store, which allows it to form a more complete picture of users' ad exposures, across its own properties and a substantial proportion of other non-Google websites.").

17. In Google's earliest years, over 99 percent of its revenues were generated by its advertising business[18] and, in 2020, Google continued to generate over 80 percent of its total revenues from advertising.[19] Google's development of the Android operating system for mobile devices complements its advertising business because Google collects information about users, their devices, and their interactions with apps every time an app is installed or updated on a Google Android device.[20] In addition, as consumers started to spend more time on mobile devices as compared to laptops and PCs, Google needed to ensure that Google's services reached mobile device users in order to collect user data. Accordingly, Google invested in Google Android so that these consumers would use its proprietary GMS suite of apps and interfaces.[21] Consumers' use of these services provided similarly valuable user data from mobile devices as that provided by Google's proprietary software on PCs and laptops.

OVERVIEW OF ANALYSIS

18. Google Android has attained a share of over 99 percent of the market for licensed mobile device operating systems.[22] For OEMs that have manufactured mobile devices, Google Android is the licensed operating system of choice. These OEMs cannot install Apple's iOS on their mobile devices because Apple refuses to license its operating system, preferring to manufacture its own devices. And since apps are neither interoperable nor transferable across the Android and iOS systems, there exist distinct markets for app distribution within each of the two operating systems. Thus, once a consumer has selected the Google Android mobile device ecosystem through the purchase of an initial device, the consumer is "locked-in" to the Google Android ecosystem, and Apple does not meaningfully constrain Google's pricing for the distribution of Android Apps.[23]

19. To pre-install the Google Mobile Services ("GMS") suite of Apps and interfaces on their own devices, OEMs must enter into a licensing agreement with Google. The Play Store, formerly known as Android Marketplace, is part of Google Mobile Services and is an "app store" that provides consumers with a range of Apps they can download and use on their Android devices. An app store is a two-sided platform: on one side, developers offer apps for download and purchase, while on the other side, consumers search for and purchase apps to download to their devices. Such two-sided platforms are characterized by what economists call "indirect network effects," meaning that the value of the platform to the users on one side is increased when there are more users on the other side of the platform. Here, the value of an app store to consumers is increased when more developers offer more apps on the platform. In turn, the value of an app store to developers is increased when there are more consumers utilizing the store to search for and

---

18. Google SEC Form 10-K, for the fiscal year ended Dec. 31, 2007, at 39, *available at* www.sec.gov/Archives/edgar/data/1288776/000119312508032690/d10k.htm.

19. Google SEC Form 10-K, for the fiscal year ended Dec. 31, 2020, at 10, *available at* www.sec.gov/Archives/edgar/data/1652044/000165204421000010/goog-20201231.htm.

20. Google Privacy & Terms, *Google Privacy Policy*, available at policies.google.com/privacy?hl=en-US.

21. Android, *The best of Google, right on your devices*, *available at* www.android.com/gms/ ("Google Mobile Services (GMS) is a collection of Google applications and APIs that help support functionality across devices.").

22. *See* Statcounter, *Mobile Operating System Market Share Worldwide*, (accessed Feb. 2022), available at gs.statcounter.com/os-market-share/mobile/worldwide/#yearly-2019-2019-bar (Android and iOS shares at 69.74% and 29.49%, respectively, as of February 2022); *Statista Mobile OS Shares* (Android and iOS shares at 72.84% and 26.34%, respectively, as of June 2021).

23. Although consumers can switch to different ecosystem when purchasing a new phone, competition in that dimension is insufficient to constrain Google from imposing anticompetitive take rates on developers.

download apps. Indeed, Google's internal documents and depositions taken to date reveal that consumer reach—the number of consumers utilizing an app store for initial downloads and purchases—is a critically important factor in the Play Store's attractiveness to developers.[24]

20. In October 2008, when Google launched its Android app store, Google's primary goal was to ensure adoption of Google Android by the array of OEMs and cellular carriers that previously provided their own branded devices with differentiated software and operating systems[25]—and by extension, by consumers choosing Android devices over Apple and Blackberry devices. By diverting a significant portion of the funds earned from its take rate charged to developers as well as advertising revenues to mobile carriers (and some OEMs abroad), Google dissuaded those carriers and OEMs abroad from establishing or pre-installing on these devices any rival app stores that might have competed with the Play Store.

21. Accordingly, while Google has consistently allowed Android App developers to keep 70 percent of the proceeds of all App sales and the sale of In-App Content, the distribution of the remaining 30 percent has changed over time. As shown in Figure 1 below, Google originally retained, at most, five percent of developer revenue as compensation for Play Store services, diverting 25 percent of developer revenue to mobile carriers (in the United States) and OEMs (mostly for devices abroad)[26] as an incentive to entrench the Play Store on more mobile devices and to discourage the initiation and success of competing app stores. Google's witnesses have testified that this allocation resulted in a loss for Google on a per-transaction basis,[27] a loss Google absorbed to convince mobile carriers, potential competitors, not to compete in the Android App Distribution Market. Once the Play Store attained significant penetration and user adoption, Google reduced the carriers' share of the take rate and ultimately terminated its revenue-sharing agreements with the mobile carriers.

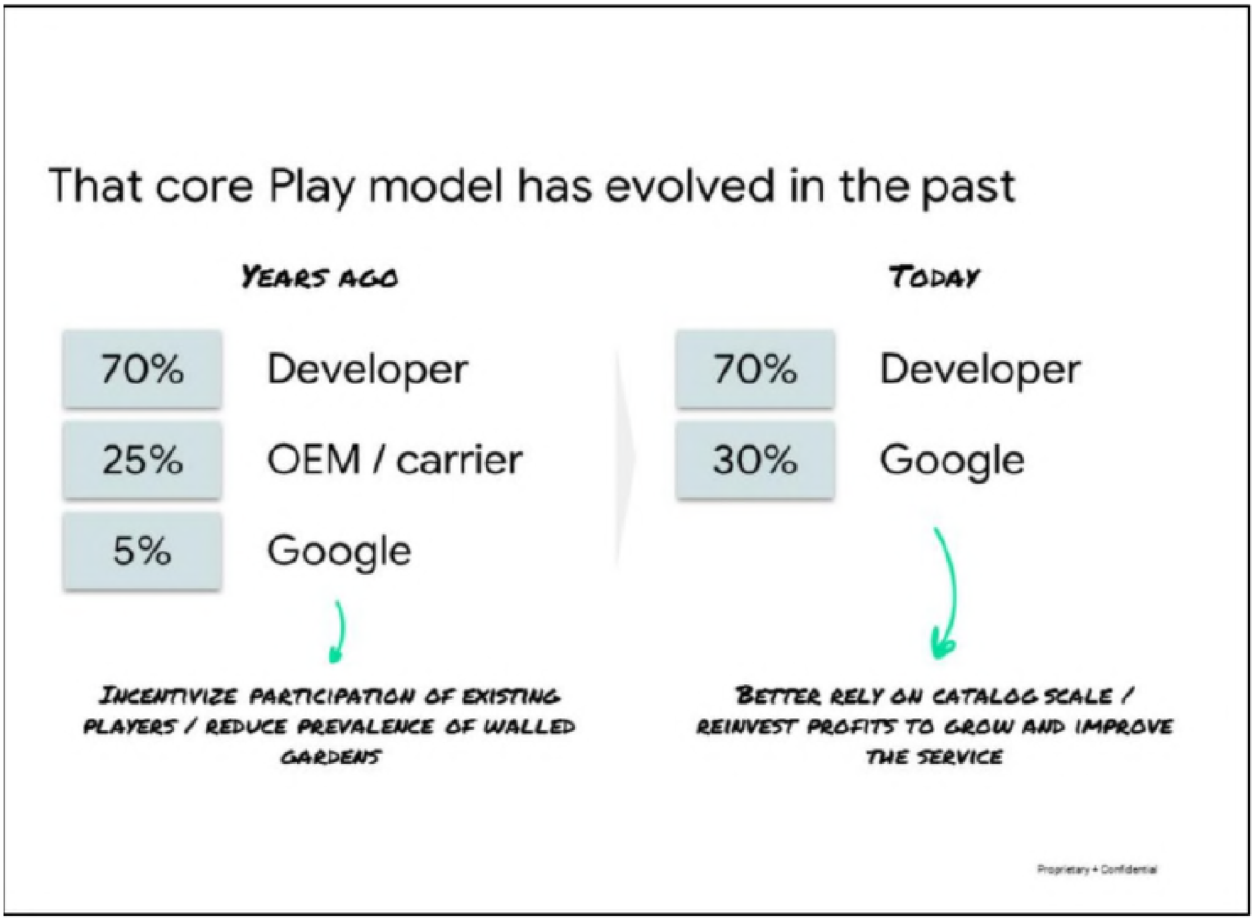
24. GOOG-PLAY-007317611 at GOOG-PLAY-007317614 and GOOG-PLAY-007317615 (Report of Dr. Itamar Simonson, Feb. 8, 2016) ("[T]he demand (or expected demand) for and profitability of developing applications for a platform are by far the most important drivers of decisions to develop applications for a platform."). *See also* Deposition of Lawrence Koh (Dec. 9, 2021) [hereafter Koh Dep.] at 321:19-323:1; 324:6-12 (the developers [redacted] and [redacted] decided not to distribute on other Android app platforms because "there was engineering work required for us to be able to utilize those services" and because of the size of Play's distribution platform across Android devices).

25. *See* Part III.D.2, *infra*.

26. Google entered revenue-sharing agreements with OEMs and carriers based on the party that owned the "client ID" for a given phone. In the United States, the carriers control the device's client ID in nearly all cases, and accordingly receive revenue sharing. GOOG-PLAY-007847148 (Deposition of Jamie Rosenberg (July 14, 2020) in *In re Google Antitrust Litigation*) [hereafter Rosenberg Dep.] at 126:10-129:9.

27. Deposition of Eric Chu (Dec. 20, 2021) [hereafter Chu Dep.] at 84:10-88:7 ("As I said, I was focusing on locking down 70 percent developers, 25 percent for carriers. I also said that I know factually at 5 percent Google would be losing money on both the transaction and also running the store.").

FIGURE 1: SPLIT BETWEEN GOOGLE AND MOBILE CARRIERS/OEMS OVER TIME



Source: GOOG-PLAY-000443763 at GOOG-PLAY-000443772.

The Android App Distribution Market

22. The Play Store is Google's two-sided platform for bringing together developers and consumers, allowing developers to sell and distribute Android-compatible Apps and consumers to purchase Apps for use on their Android devices. In addition to providing matchmaking between consumers and developers, other functions in the Android App Distribution Market include but are not limited to auto-updating and storage.[28] With the exception of China, where the Play Store is blocked,[29] "Apple and Google control more than 95 percent of the app store market share through iOS and Android…The app economy was built on these two platforms[.]"[30] Due in part to the massive installed base of Android mobile devices, the Play Store accounts for more than

28. The take rate compensates Google for all of these functions. There is no separate demand among consumers or developers for any of these functions.

29. VPNDada, *How to Access Google Play Store in China*, *available at* www.vpndada.com/access-google-play-store-china/ ("If you buy an Android phone in China today, you won't find the Google Play app store pre-installed on that phone. Instead, depending on the brand of the phone, it will come with some other app stores, mostly likely one offered by a Chinese company.").

30. David Curry, *App Store Data (2022), Business of Apps*, (Jan. 11, 2022), *available at* www.businessofapps.com/data/app-stores/ ("Outside of China, Apple and Google control more than 95 percent of the app store market share through iOS and Android, respectively…The app economy was built on these two platforms, which have expanded their offerings to include apps for consumers and every type of business."). *See also* Part II.C.2.

three times as many downloads as the Apple App Store worldwide—despite the Play Store's absence from China.[31]

23. The Android App Distribution Market is a relevant product market that is distinct not only from Apple's iOS app distribution market, but also from the markets for web-based apps and distribution channels for applications for PCs or gaming consoles. Given the widespread distribution of the Play Store throughout the world, developers of Android-compatible Apps, wherever they are located, have strong incentives to list their Apps for distribution on the Play Store. The global reach of the Play Store and the developers who seek to distribute their Apps through it thus makes the relevant geographic market for the Android App Distribution Market global, excepting China, where the government prefers Chinese providers of both mobile devices and operating systems.[32]

24. Direct and indirect evidence establishes that Google has market power in the Android App Distribution Market. Google's 30 percent take rate is high relative to competitive benchmarks, yet the vast majority of apps are downloaded through the Play Store. As a two-sided platform, the Play Store benefits from indirect network effects, which serve to entrench its market share with developers. Given Google's reach with consumers gained and maintained through the restrictions, and the anticompetitive conduct that has substantially foreclosed alternative channels for Android-compatible Apps, developers effectively must list their Apps on the Play Store and agree to its restrictive conditions, which in turn act as substantial barriers to entry for effective competition from rival app stores.

25. Google has provided inducements, imposed a variety of restrictions, and erected various technological barriers to substantially foreclose rival app stores and the direct downloading of Apps. It has done so to achieve and maintain its market power in the Android App Distribution Market. More specifically, Google has engaged in the following conduct in the Android App Distribution Market:

a. *Financial Inducements*: Google achieved its power in the Android App Distribution Market by making payments that incentivized carriers (and OEMs) to distribute Google Android mobile devices and dissuaded (and in some cases prevented) them from developing, promoting, or offering alternative app stores, including their own stores. Google was willing to operate the Play Store at a loss to achieve these purposes but, once power was achieved, Google dramatically reduced or eliminated these payments.

b. *Bundling of Apps and APIs*: Google requires OEMs to pre-install and prominently place the Play Store on all Google Android devices as a contractual

31. *See, e.g.,* Sensor Tower, *2021 – 2025 Mobile Market Forecast,* (2021) at 7, *available at* go.sensortower.com/rs/351-RWH-315/images/Sensor-Tower-2021-2025-Market-Forecast.pdf (showing 109 billion Play Store app downloads worldwide in 2020, compared with 34 billion app downloads in the Apple App Store). Industry data show that the Play Store also accounts for approximately 90 percent of non-Apple, non-China mobile device app expenditures worldwide *See* Part II.C.2 for derivation.

32. GOOG-PLAY-004253884 at GOOG-PLAY-004253894 ("Samsung Leads in All Retail Driven Markets except China") (chart shows Apple and Samsung shares in different countries, with China being dominated (60 percent) with local OEMs). It bears noting that all references to "global" markets or the use of the terms "globally" or "world" in this report assume that China is excluded.

condition of licensing GMS, which includes Google's most popular Apps, including Google Maps, YouTube, Chrome, Google Search, and Gmail. OEMs must also install this bundle of apps to gain access to crucial programming interfaces necessary for many common android apps to properly function. The Play Store's prominent position inhibits competition from competing app stores.

c. *Anti-steering Restrictions*: Google's agreements with App developers prohibit developers from steering users within the App to other app stores, platforms, or websites to purchase or download Apps.

d. *Tie of YouTube, Google Search and Play Store Advertising to the Play Store*: App developers' access to valuable advertising opportunities on YouTube and Google Search is conditioned on the sale and distribution of developers' Apps through the Play Store.

e. *Technical Barriers*: Google imposes default settings and warnings that make it unnecessarily difficult for users to download rival app stores and Apps from rival app stores or from developer websites, and Google limits auto-updating functionality to the Play Store and certain pre-installed Apps.

26. These restraints inhibit the development of alternative competing app stores and the use of direct downloads from developers' websites to install apps on Google Android devices. Multi-homing, the use of alternative app stores on the same device, would occur more extensively in the absence of Google's restraints. In a competitive world, steering via altering the relative price of initial downloads of Apps would allow developers to direct consumers to lower-priced alternatives, including direct downloads from their own websites or competing app stores that charge lower take rates. With a combination of multi-homing and steering, developers could charge a relatively lower price for Apps to consumers who download Apps from a lower-cost app platform or website. This would in turn exert competitive pressure on Google to lower its own take rate.

The Aftermarket for Services in Support of Consummating Purchases of In-App Content

27. Following the download and installation of an App, developers may continue to offer digital content to the consumer related to the App. The matchmaking services offered in Android App Distribution Market are distinct from those services offered in support of consummating purchases of In-App Content. Unlike the two-sided nature of the Android App Distribution Market, the In-App Aftermarket is one-sided: It is a simple transaction between a buyer (the developer) and a seller of services, including payment processing, record keeping, and unlocking of content, needed to consummate a purchase of In-App Content. In economic parlance, indirect network effects are not present in the In-App Aftermarket. From the developer's perspective, certain functions are needed for a consumer to be able to purchase In-App Content, including billing (also present in the Android App Distribution Market) and unlocking the In-App Content on the user's phone (not present in the Android App Distribution Market). The developer's demand for these services in the In-App Aftermarket is derived from the demand by the consumer for the In-App Content itself. While the App developer delivers the In-App Content, because of Google's requirements, an App developer cannot complete the transaction without using Google's services, mainly Google Play Billing. By forcing developers to complete transactions through

Google's payment and unlocking system, Google Play Billing, Google has effectively tied the In-App Aftermarket to the Android App Distribution Market and forced developers who distribute an App to a consumer through Google to forever use Google as a middleman for the consumers' purchase of In-App Content by virtue of its Google Play Billing requirements.[33] Absent this tie-in, developers could either provide or engage third parties to provide the services in the In-App Aftermarket, now provided by Google.

28. Once an App is purchased and downloaded from the Play Store, the Play Store need not play any role in the In-App Aftermarket. Through the Android App Distribution Market, the platform has brought the developer and the consumer together and facilitated the download and installation of the App on the device. I understand that Plaintiffs' technical expert, Professor Douglas Schmidt, has found that, although Google inserts itself into the In-App Aftermarket by requiring that developers use Google Play Billing, there is no technical justification for requiring developers to use Google Play Billing. Similarly, while Google requires that every App downloaded through the Play Store utilize the services that Google has included in its Google Play Billing product for the sale of In-App Content, there are numerous alternatives that can provide similar features at a lower cost to developers, and ultimately consumers.

29. Google maintains market power in the In-App Aftermarket by requiring developers to use Google Play Billing to support the purchase of all In-App Content. Google utilizes Google Play Billing to impose a take rate generally of 30 percent—the same take rate it commands in the Android App Distribution Market—on all purchases of In-App Content, forever. The In-App Aftermarket is a distinct relevant market, and in the but-for world, developers could select many potential competitors to support the purchase of In-App Content. This is evidenced by Google's prohibition on developer steering of consumers to outside channels, the ability of more powerful developers to bypass Google Play Billing, and Google's efforts to incentivize these developers to transact through Google Play Billing. The geographic In-App Aftermarket is global, except for China, as third party In-App Aftermarket service providers could provide cross-border global services.

30. Direct and indirect evidence establish Google's market power in the In-App Aftermarket. Google's standard 30 percent take rate is high compared to rates charged by potential competitors, yet 97 percent of all developers offering In-App Content utilize Google Play Billing by virtue of Google's restraint.[34] Moreover, Google routinely discriminates in price among developers, not requiring those selling in-app physical goods to utilize Google Play Billing, and, more recently, reducing its take rate on subscription sales or sales by smaller developers, reflecting

33. Once a payment is consummated, the In-App Content is either unlocked within the App itself or accessed and downloaded by the App from an independent server. In addition to payment processing, other services currently performed by Google in the In-App Aftermarket (by virtue of its exclusionary restrictions) include record keeping, server hosting, and security.

34. Record evidence indicates Google recognized early on that its 30 percent take rate was difficult to justify. *See, e.g.,* GOOG-PLAY-004338990 (Internal December 2009 e-mail seeking "[e]xact articulation of under what circumstances transactions *we* [Google] now solely take 30%... I don't have any idea how much money we think this will make for us, but it will cost us a lot of good will. Further, if Android is here to increase phone usage and thus web usage and thus drive ad revenue, we need to give developers every penny we reasonably can…With Passion we are apparently keeping 30% on all transactions, I can't imagine transaction processing costs us that much. Why don't we take this opportunity to let developers keep more revenue? I don't know why this change has been made and there is deafening silence from those who know. The only reasoning I can come up with makes no sense, which is even worse than reasoning I disagree with.").

the fact that Google faces a downward-sloping demand curve among developers, a hallmark of market power.[35]

31. Google has maintained its market power in the In-App Aftermarket by imposing a variety of restrictions and offering targeted financial incentives. More specifically, Google has engaged in the following conduct in the In-App Aftermarket:

a. *Linking Play Store Access*: A developer can offer its App for sale or distribution through the Play Store—a monopolist in the Android App Distribution Market—only if the developer agrees to exclusively use Google's in-app system, Google Play Billing, for all subsequent sales of In-App Content.

b. *Anti-Steering Restrictions*: Google contractually prohibits developers from steering customers within the App to alternative distribution and payment processing outlets for purchasing In-App Content outside the Play Store and prohibits them from even using any customer information the developer learned through the Play Store.

c. *Targeted Incentives*: Google provides large monetary incentives and advertising packages to ensure compliance by those developers that have the resources to create alternative app stores or billing systems.

32. For ease of exposition, I refer to this collection of restraints as the "Aftermarket Restrictions," and I refer to the first restriction (a) in particular as the "Aftermarket Tie-in" or "Tie-in."[36] In the absence of the Aftermarket Restrictions, competition in the In-App Aftermarket would be robust. Developers would be able to select their own suppliers, or they would offer consumers the option of selecting a payment processing system, as well as other features (e.g., record keeping, server hosting, and security) provided in the In-App Aftermarket, from an array of competitive options and could steer consumers towards lower-priced alternatives.

The Impact of Google's Anticompetitive Conduct on the Class of Consumer Plaintiffs

33. Google's anticompetitive restrictions in the Android App Distribution Market and in the In-App Aftermarket have impaired competition in both markets. These anticompetitive restrictions have damaged the Consumer Plaintiffs by raising the prices of paid Apps and In-App Content for all or nearly all members of the class of Consumer Plaintiffs (the "Class"). Absent Google's restrictions and anticompetitive conduct, competition in the distribution of applications would materialize, and developers would likewise have a choice of how to transact their In-App Content with their own users. The benefits to consumers resulting from competition could take different forms. I use a two-sided platform model with multi-homing to show that, in the absence of restraints in the Android App Distribution Market, Google would be compelled to lower its take rate from developers. I show that a portion of the savings to developers from a take rate reduction

---

35. Firms that lack market power, by contrast, face a horizontal demand curve, which means they cannot restrict output by raising prices.

36. Although the last item c can also be understood as an incentive, when employed by a firm with monopoly power, a loyalty rebate or bundled loyalty discount can foreclose rivals and thereby serve as a restraint on trade. *See, e.g.*, Patrick Greenlee, David Reitman, and David S. Sibley, *An antitrust analysis of bundled loyalty discounts*, 26 INTERNATIONAL JOURNAL OF INDUSTRIAL ORGANIZATION 1132-1152, 1132, 1135, 1137-38 (2007).

would be reflected in lower consumer prices. Next, I show that Google would have responded to greater distribution competition by increasing consumer subsidies through, for example, its Play Points Program.[37] Consumers would obtain the benefits of these subsidies directly from Google. Finally, I use a single-sided model to show that, in the absence of its Aftermarket Restrictions, Google would be compelled to respond to competition by dropping its take rate in the In-App Aftermarket, which would result in lower prices to consumers.

34. Using data, economic methods, and evidence common to all Class members, I demonstrate how aggregate damages and antitrust injury attributable to Google's anticompetitive conduct can be assessed through a reduced take rate or increased consumer Play Points subsidies. I also demonstrate how common methods and evidence can be used to measure specific Class-member damages, based on the 35 categories of Apps that Google uses to track user purchase activity.

35. The remainder of the report is organized as follows. In Part I, I explain why Apple's iOS does not constrain Google's market power in the Android App Distribution Market or the Android In-App Aftermarket. In Part II, I explain why these two markets are distinct relevant antitrust product markets. In Part III, I examine Google's anticompetitive conduct in the Android App Distribution Market, defining the contours of the relevant market, establishing Google's market power in that market, describing the restraints imposed by Google, and describing how multi-homing and steering would combine to lower Google's take rate in the Android App Distribution Market. In Part IV, I perform the same analysis for the In-App Aftermarket and additionally explain why the Single Monopoly Profit theory does not apply, and therefore Google is obtaining additional profits from leveraging its power into the In-App Aftermarket that it could not have obtained from the Android App Distribution Market alone. In Part V, I analyze and assess the impact of Google's anticompetitive conduct in the relevant markets. In Parts VI and VII, I estimate damages in the aggregate and at the individual level, respectively. The analysis in all these parts relies on methods and evidence common to members of the proposed class.

I. APPLE'S IOS DOES NOT CONSTRAIN GOOGLE'S MARKET POWER IN THE ANDROID APP DISTRIBUTION MARKET OR IN-APP AFTERMARKET

36. Mobile device OEMs must either develop their own operating system or license an operating system from a third party. Apple is the only significant device manufacturer to develop and maintain its own operating system, iOS, which works exclusively on Apple devices. For all other OEMs, Google is the dominant provider of licensed mobile device operating systems. Google Android OS makes up nearly all licensed mobile operating systems, 73 percent of mobile operating systems worldwide (even including China),[38] and 40 percent of mobile operating systems in the United States.[39]

---

37. As explained below, Play Points is a customer loyalty program implemented by Google, similar to those frequently offered by credit card issuers and airlines. I use the term "Play Points" throughout this report as shorthand for promotions funded by Google that decrease the net price that consumers pay for Apps and In-App Content.

38. *Statista Mobile OS Shares, supra*. Statista does not break out China separately. Android and iOS collectively account for 99 percent share as of June 2021. *Id.*

39. Statcounter, *Mobile Operating System Market Share United States of America*, (accessed Feb. 4, 2022), *available at* gs.statcounter.com/os-market-share/mobile/united-states-of-america.

A. Google Android and Apple iOS Have Market Power in the Mobile Device Operating Systems Market

37. At present, there is a virtual duopoly in the market for mobile device operating systems with the market split between mobile devices running on Apple's iOS and Google Android.[40] The combined share of Android and iOS mobile devices is estimated to exceed 99 percent.[41] There are several minor mobile operating systems, but collectively they account for less than one percent of the worldwide market. While Android accounts for over 70 percent of the worldwide market, mobile devices running iOS command nearly 60 percent of sales in the United States.[42] Industry analysts attribute the difference in market penetration to the higher cost of the iPhone, which is beyond the means of many non-U.S. residents.[43]

38. Even firms with desktop operating systems such as Microsoft have failed to make significant inroads into mobile device operating systems. In January 2019, Microsoft announced that it would end support for its Windows 10 mobile operating system.[44] Microsoft's failure is consistent with significant barriers to entry in mobile operating systems markets.

B. While Android Mobile Devices May Compete at the Point of Initial Device Purchase, Lock-In Results in Distinct Markets for App Distribution

39. Google Android mobile devices and Apple compete for users with respect to the initial choice of a device and associated ecosystem. But once a user selects either a Google Android or iOS device, that user is largely locked in. Because apps are neither interoperable nor transferable, a user switching from an iOS device to a Google Android device or vice versa cannot simply bring all apps to the new phone. The user must re-download and install apps, and may need to find substitutes for apps not available on the new OS. Users may need to repurchase paid apps and any in-app content, and they may end up losing their app-related data if unable to transfer.[45] Transferable app data and customization settings must be transferred to the new device, a process

---

40. *Statista Mobile OS Shares*, *supra*. For purposes of this class certification report, I include tablets in the market for mobile device operating systems, as OEMs (the buyer of mobile operating systems) selling both tablets and smartphones prefer that the operating system work seamlessly across mobile device types. *See, e.g.*, Vangie Beal, *What Are Examples of Mobile Operating Systems?*, Webopedia (Jan. 27, 2022), *available at* www.webopedia.com/insights/mobile-os-and-different-types/. The OEM's demand for compatibility in operating systems across the device types is derived from the demand from consumers, who share the same preferences. *See* Part I.B, *supra*, discussing customer lock-in for operating systems. When performing econometrics and estimating damages, I also include App transactions on tablets in Google's transactional database. In any event, my opinions regarding Google's market power or common impact flowing from the Challenged Conduct do not turn on whether tablets are included or excluded.

41. *Statista Mobile OS Shares*, *supra*. This estimate includes China.

42. *See* Jack Wallen, *Why is Android more popular globally, while iOS rules the US?*, TECH REPUBLIC, (May 12, 2021), *available at* www.techrepublic.com/article/why-is-android-more-popular-globally-while-ios-rules-the-us/.

43. *Id.*

44. *See* Rob Enderle, *How Microsoft failed with Windows 10 Mobile*, COMPUTERWORLD, (Jan. 24, 2019), *available at* www.computerworld.com/article/3336057/how-microsoft-failed-with-windows-10-mobile.html.

45. Nabila Amarsy, *Switching Costs: 6 Ways to Lock Customers Into Your Ecosystem*, STRATEGYZER, (July 27, 2015), *available at* www.strategyzer.com/blog/posts/2015/7/27/switching-costs-6-strategies-to-lock-customers-in-your-ecosystem ("The 'Data trap'… encourages customers to create or purchase content or apps that are exclusively hosted on a platform.").

that entails additional time and effort.[46] Likewise, subscriptions might need to be cancelled and re-subscribed when switching platforms.[47]

40. Google's internal documents provide evidence of consumer switching costs. A 2017 presentation titled "Switching to Pixel" describes the "switching user journey."[48] User concerns are captured under the general heading "Fear of the Unknown," which posits questions users may ask before switching from an iPhone to an Android device (including but not limited to Google Pixel), including, "Will it do everything my current phone does?" "Will I be able to access all my data?" "Will I know how to use it?" "What will it say about me?"[49] The presentation reviews the multi-step "end-to-end switching experience."[50] Other Google documents identify "losing data," "learning a new OS, "apps/feature incompatibility" and "phone spec inequality" as the "biggest concerns" users have associated with switching.[51] Google's analysis from 2017 concluded that "switching is not as easy as we think."[52]

41. Once someone is accustomed to using a brand, there can be identity and loyalty factors inhibiting changing to another,[53] raising the costs of switching. These various switching costs—reinstallation and substitution, transferring data, re-subscribing, and brand loyalty—are also all paired with the tedious process of learning the new operating system.[54] The Play Store allows Google to reach users who then download Android Apps, which creates brand loyalty to Android. Indeed, Google recognizes that the Play Store "is creating brand loyalty and 'stickiness' to Android and the Google ecosystem thus becoming a critical channel to our end users."[55]

42. Users also benefit from the complementarity of multiple devices; switching ecosystems often requires relinquishing those benefits. For instance, a user might use her smartphone when traveling during the day and switch to using a tablet later at night. This user would likely gain value from the ability to sync files, settings, user information, and other features among these multiple devices.[56] Yet many apps do not allow for this synchronization across

46. *Id.*

47. Economists recognize that firms in ancillary markets or aftermarkets may wield power *even when* the forward market is competitively supplied. For a review of the literature, see Hal Singer & Andrew Card, *Lessons from Kahneman's Thinking Fast and Slow: Does Behavioral Economics Have a Role in Antitrust Analysis?*, ANTITRUST SOURCE (2012), *available at* www.semanticscholar.org/paper/Lessons-from-Kahneman-%E2%80%99-s-Thinking-%2C-Fast-and-Slow-Kahneman/8abf422dc2aca5adf6fe6c20e9064863f64819dd?p2df. As noted in Part I.A, Apple does not constrain Google in the forward market for licensed mobile operating systems, and thus the forward market is hardly competitively supplied.

48. GOOG-PLAY-007317466 at GOOG-PLAY-007317467. Pixel is a Google smartphone.

49. *Id.* at GOOG-PLAY-007317473.

50. *Id.* at GOOG-PLAY-007317479.

51. GOOG-PLAY-000880576.R at GOOG-PLAY-000880580.R.

52. *Id.* at GOOG-PLAY-000880584.R. Google's analysis concerned users switching from iOS to Google's Pixel phone. It further concluded that switching "feels a lot like learning a foreign language." *Id.* at GOOG-PLAY-000880589.R.

53. *See* Thomas A. Burnham, Judy K. Frels, and Vijay Mahajan, *Consumer Switching Costs: A Typology, Antecedents, and Consequences*, 31(2) JOURNAL OF THE ACADEMY OF MARKETING SCIENCE 109-126 (2003).

54. Older users have spent more time with Android devices, and the switching costs appear to accumulate with time invested in the Android device. *See, e.g.*, GOOG-PLAY-002416488 (Spotify data showing users under 24 years old switch to iOS more frequently than older users).

55. GOOG-PLAY-004237669.R at GOOG-PLAY-004237673.R.

56. Different operating systems offer different features for synchronizing devices. For instance, Microsoft Windows offers a "Your Phone" app which can tie together an android operated phone and computer. Similarly, Apple

different operating systems. Accordingly, effective transfer to the Google Android or Apple ecosystem may entail switching multiple devices.

43. This complementarity also takes place within groups of people. For instance, a family of four that is contemplating purchasing multiple devices would value the ability to integrate their apps and files—perhaps through parental controls on apps and screen time, location tracking abilities, and through family plans allowing the sharing of purchases and subscriptions.[57] To achieve this compatibility, the family would likely need to all use the same operating system, leading them to become "locked in" to that ecosystem.[58] This type of lock-in can also lead to brand loyalty.[59] That consumer lock-in occurs with the initial decision with respect to handsets (and the associated mobile operating system) is consistent with lock-in in other contexts and is widely noted in the economics literature.[60]

44. Consumer lock-in is precisely what happens for users who choose a Google Android phone (many of whom outside of the United States cannot afford an iPhone). The average holding period for an Android phone is typically two years, or longer than the time required to test for a "sustained" price increase by a hypothetical monopolist under the test for market definition set forth by the United States Department of Justice and Federal Trade Commission in their *Horizontal Merger Guidelines*. Moreover, Android users display a high degree of brand loyalty, at around 89 to 91 percent over the year 2017,[61] which means that even on their next phone purchase, they are unlikely to purchase an Apple iPhone, especially given its much higher price—on average 2.7 times the cost of the typical Android phone.[62] Internal Google figures corroborate this brand loyalty, finding in 2016 that Android had 86 percent brand loyalty and a 21-month

---

offers a "Continuity" app with similar features for Apple products. *See* David Nield, *Make your phone and computer team up to get more done*, POPULAR SCIENCE, (Oct. 20, 2018), *available at* www.popsci.com/phone-computer-work-together/.

57. For example, Apple's "Family Sharing" allows families to share services such as Apple Music, Apple TV+, App Store purchases, iCloud storage, and photo albums. It also allows parents to approve what their children purchase, limit their time on devices, and see their location. *See* Apple, *What is Family Sharing?, available at* support.apple.com/en-us/HT201060. *See also* giffgaff, *How to set up family sharing on iPhone and Android*, (July 26, 2019), *available at* www.giffgaff.com/blog/how-to-set-up-family-sharing-on-iphone-amp-android/ (describing the process of setting up the Google Play Family Library service).

58. Travers Korch*, How tech ecosystems lock you into costs*, BANKRATE, (Sep. 30, 2014), *available at* web.archive.org/web/20210602055404/https:/www.bankrate.com/finance/smart-spending/tech-ecosystems-cost-you-1.aspx%20.

59. Paul Klemperer, *Markets with Consumer Switching Costs*, 102(2) QUARTERLY JOURNAL OF ECONOMICS 375-394, 376 (1987) ("In all these markets [with switching costs] rational consumers display brand loyalty when faced with a choice between functionally identical products. Products that are ex ante homogeneous become, after the purchase of one of them, ex post heterogeneous").

60. *See, e.g.,* Joseph Farrell & Paul Klemperer, *Coordination and Lock-In: Competition with Switching Costs and Network Effects* 3 HANDBOOK OF INDUSTRIAL ORGANIZATION 1970-2056, 1970 (2007) ("Lock-in hinders customers from changing suppliers in response to (*predictable or unpredictable*) changes in efficiency, and gives vendors lucrative ex post market power – over the same buyer in the case of switching costs (or brand loyalty), or over others with network effects.") (emphasis added).

61. Lucas Mearian, *iOS vs. Android: When it comes to brand loyalty, Android wins*, COMPUTERWORLD, (Mar. 9, 2018), *available at* www.computerworld.com/article/3262051/ios-vs-android-when-it-comes-to-brand-loyalty-android-wins.html.

62. Amit Chowdhry, *Average iPhone Price Increases To $687 and Android Decreases To $254, Says Report*, FORBES, (Feb. 3, 2015), *available at* www.forbes.com/sites/amitchowdhry/2015/02/03/average-iphone-price-increases-to-687-and-android-decreases-to-254-says-report/?sh=d9bcd17539e4.

purchase cycle.[63] Google marketing analytics in 2018 and 2019 found that [redacted] percent of Android owners purchasing a new device within the previous twelve months purchased another Android device.[64] Because lock-in increases with each successive year on an Android phone, the share of Android users willing to even *consider* switching to an iPhone is falling over time.[65]

45. Because apps are operating-system specific, Apple users cannot obtain apps from the Play Store, and Android users cannot download apps from the Apple App Store. Accordingly, while Google and Apple undoubtedly compete for users in the adoption of devices powered by their respective operating systems, once a user has elected the Android ecosystem, Apple's App Store cannot provide that user with apps and cannot compete with Google in the distribution of apps.[66] Further, the presence of switching costs and the lack of information that consumers receive at the point of device purchase[67] allows Google to extract supracompetitive profits from developers and consumers alike, as iOS does not serve as a significant constraint on Google's exercise of its monopoly power in the Android App Distribution Market.

C. Google Has Monopoly Power in the Distinct Market for Licensed Operating Systems

46. Apple does not license its iOS to any other OEM. Thus, from an OEM's vantage, iOS is not a substitute for licensed mobile operating systems. In terms of the test for market definition in the *Horizontal Merger Guidelines*, a small, but significant, non-transitory increase in price (a "SSNIP") by a hypothetical monopolist of licensed operating systems could not induce any OEM to install iOS instead, as Apple will not grant such a license. Moreover, the costs and complexities of developing an operating system are so high that a SSNIP by a hypothetical monopolist of licensed operating systems would not plausibly induce an OEM to create its own system or a rival mobile OS developer to enter. Google dominates the market for licensed mobile operating systems through its Android OS, with nearly a 100 percent share of this market. Herbert Hovenkamp, the co-author of a leading antitrust treatise, recently observed, "[D]igital markets are

---

63. GOOG-PLAY-000572041.R at GOOG-PLAY-000572048.R.

64. GOOG-PLAY-004556784 at GOOG-PLAY-004556793 (Q3 2018 marketing analytics); GOOG-PLAY-005705974 at GOOG-PLAY-005705985 (Q4 2019 marketing analytics).

65. *See, e.g.*, Abhin Mahipal, *Survey: 18% of Android users would consider switching to iPhone 13, but this is down 15% from last year*, SELLCELL, (Aug. 31, 2021), *available at* www.sellcell.com/blog/survey-18-percent-of-android-users-would-consider-switching-to-iphone-13/. It bears noting that the mere *consideration* of switching does not imply an actual switch, as consideration does not entail any switching costs. In other words, although the survey states only 18 percent of Android users would consider switching, not all of those respondents would in actuality switch.

66. The court in *Epic v. Apple* found that the relevant market for Epic's gaming-centric complaint was the "mobile gaming market," which included iOS and Android mobile devices. There the court followed the factors from *Newcal Indus., Inc. v. Ikon Office Sol.*, 513 F.3d 1038, 1049-50 (9th Cir. 2008), which are divorced from economic teachings, to reject a single-product aftermarket. In particular, there is no economic requirement that consumers be duped in order to be locked into the Android operating system and then beholden to a provider in the Android App Distribution Market. Lock-in is not an information problem. Movie patrons who pay supra-competitive prices for popcorn, or hotel guests who pay supra-competitive prices for movie rentals are not duped; they simply do not wish to incur the costs of leaving the movie theater (or hotel) to find a cheaper substitute for the ancillary product. In any event, Play Store users have no understanding of the mapping from restraints into developer take rates into prices; to the extent *Newcal* (improperly) requires an information problem for the existence of an aftermarket, it is here in spades.

67. For instance, consumers are not told that developers are generally charged 30 percent for every paid app or in-app purchase, nor are they told of Google's restrictions on OEMs, carriers, and developers that result in foreclosure of competitive application stores. Instead, they have historically been told that Google's Android is an "open" system. Without knowledge of the anticompetitive restrictions, there is no way for consumers to know that they will effectively be locked into a product that extracts supracompetitive profits every time they make a purchase.

particularly susceptible to direct measurements of market power that do not depend on a market definition."[68] Such "direct" proof relies on firm-specific information that speaks directly to a firm's ability to profitably raise prices or exclude rivals.[69] Google's monopoly in licensed operating systems is well-established and is demonstrated through both direct and indirect methods.

1. Direct Evidence

47. Google gained power in the licensed mobile device operating system through its acquisition of Android and partnerships with OEMs and mobile carriers. OEMs could not license Apple's iOS to manufacture mobile devices, but Google's Android OS and its proprietary suite of mobile apps and interfaces, Google Mobile Services, was available under license. While Android itself is available pursuant to an open-source Apache license, Google used the revenue-sharing agreements to attract OEMs and mobile carriers to select Google Android. In its revenue-sharing agreements, Google offered OEMs and mobile carriers significant percentages of the revenues generated by Google's search advertising and the Play Store.[70] The ability to exclude rivals—here, rival licensable operating systems that lacked a dominant search advertising business with which to fund OEMs and mobile carriers—is a hallmark of market power.[71]

2. Indirect Evidence

48. Google's high share of the licensed mobile device operating systems market is protected by barriers to entry. Entry barriers in the market for mobile operating systems are steep.[72] Launching a competing mobile operating system requires developing the source code, contracting with OEMs, and convincing a critical mass of developers to write apps that are compatible with the new operating system, among other hurdles.[73]

---

68. Herbert Hovenkamp, *Digital Cluster Markets*, COLUMBIA BUSINESS LAW REVIEW 1-30 (forthcoming 2022).

69. *Id.* at 24 ("By contrast, 'direct' proof relies on estimates of firm elasticity of demand, evidenced mainly by a firm's price-cost margins or output responses to price changes.[] These methodologies are capable of giving more accurate measures of market power as it is best defined, which is the ability of a firm to profit by raising its price above its costs.[]") citing 2B Phillip E. Areeda & Hebert Hovenkamp, ANTITRUST LAW ¶521 (5th ed. 2021) (forthcoming); Louis Kaplow, *Why (Ever) Define Markets?,* 124 HARVARD LAW REVIEW 437 (2010).

70. *See, e.g.*, GOOG-PLAY-001184813 at GOOG-PLAY-001184820 (showing that for 2014, Google earned $5,976M in search revenue on Android, with $467M paid to carriers and $217M paid to OEMs). By this point in time, carrier revenue shares were already reduced, suggesting the figures would be even higher in the early days.

71. *See, e.g.*, Richard G. Price, *Market Power and Monopoly Power in Antitrust Analysis*, 75(1) CORNELL LAW REVIEW 190-217, 190, 198 (1989).

72. *See, e.g.*, Alan Santillan, *Mobile Software: Why It's Only Android vs. iOS in 2020*, LEARN HUB, (July 30, 2018), *available* at learn.g2.com/android-vs-ios ("Barriers to entry in the mobile space are extremely high, and the switching costs that Android and iOS deploy toward their users make it even harder for buyers to switch platforms."); European Commission, Antitrust: Commission sends Statement of Objections to Google on Android operating system and applications – Factsheet, (Apr. 20, 2016), *available at* ec.europa.eu/commission/presscorner/detail/en/MEMO_16_1484 ("There are a number of barriers to entry that protect Google's position, including so-called network effects (that is, the more consumers adopt an operating system, the more developers write apps for that system).").

73. *See, e.g.* Michael Katz & William Rogerson, *The Applications Barrier to Entry and Its Implication for the Microsoft Remedies: Comment on Iansiti and Richards*, 75 ANTITRUST LAW REVIEW 723-738 (2008-09) (explaining that a new OS will be desirable to consumers only if a broad array of software applications can run on it, but software developers will find it profitable to create applications that run on an operating system only if there is a large existing base of users).

49. Google cemented its early market power in licensed mobile device operating systems by exploiting a natural feedback loop in which device makers adopted Android with Google Mobile Services, including Google's proprietary App distribution channel, which in turn attracted app developers, which in turn attracted consumers who sought Android phones.[74] Google Android's monopoly in the licensed mobile device operating systems market benefits from what economists call "indirect network effects"—in this case meaning that the more developers who write apps compatible with an operating system, the more OEMs and consumers demand the operating system. This cycle of indirect network effects constitutes another barrier to potential suppliers of competing licensed mobile device operating systems.[75]

II. The Android App Distribution Market and the In-App Aftermarket Are Distinct Relevant Markets

50. The purchaser of a Google Android device always receives the device with the Play Store pre-installed and its icon prominently displayed. Through the Play Store, consumers can access a broad array of Apps offered by myriad developers. Thus, the Play Store is a two-sided matchmaking platform where Google brings together developers wishing to distribute Apps and consumers wishing to obtain Apps to use on their Google Android devices. The initial download of an app may be considered the consummation of the matchmaking service, giving rise to an offer (by the developer) and acceptance (by the user). Once an App is downloaded onto the Google Android mobile device, however, the Play Store's matchmaking role for the initial download is at an end because the developer has found the consumer and created its own independent channel of distribution with that consumer by virtue of the installation of the developer's App on the consumer's device. Through its App, the developer now has a direct pipeline to the consumer for both communication and purchases of In-App Content. As discussed more fully below, the Android App Distribution Market and In-App Aftermarket are economically distinct.

51. A two-sided platform matches buyers (in this case consumers) and sellers (in this case app developers). Two-sided platforms benefit from "indirect network effects," meaning that each additional buyer makes the platform more appealing to sellers[76] Buyers wish to transact on the platform with the greatest variety of content to choose from, and sellers wish to reach the largest buyer base possible. In the context of the Android App Distribution Market, Google connects consumers of Apps with developers through the Play Store. The presence of more consumers makes the Play Store more appealing to developers, and the presence of more developers make the Play Store more appealing to consumers.[77] A two-sided platform creates

---

74. *See, e.g.,* GOOG-PLAY4-000336290.

75. *See, e.g.,* GOOG-PLAY-004559725.R at GOOG-PLAY-004559759.R (Google presentation from 2017 stating that "Play benefits from network effects. Users come to Play because we have by far the most compelling catalogue of apps/games. Developers come to Play because that's where the users are.") *See also* GOOG-PLAY-004508011 at GOOG-PLAY-004508012 (summary notes from 2019 internal Google brainstorming session stating "[a]n [A]ndroid app cannot ultimately survive as a product or business without being hosted on Play.").

76. *See, e.g.,* David Evans, *Two-Sided Market Definition* in MARKET DEFINITION IN ANTITRUST: THEORY AND CASE STUDIES (ABA Section of Antitrust Law) 1-35, 5 (2009), *available at* papers.ssrn.com/sol3/papers.cfm?abstract_id=1396751 ("A key feature of two-sided platforms is the presence of 'indirect network effects.').

77. Marc Rysman, *The Economics of Two-Sided Markets*, 23(3) JOURNAL OF ECONOMIC PERSPECTIVES 125-143, 126 (2009).

value by harnessing these indirect network effects to increase the number of participants on each side of the platform.

52. When it comes to the purchase of In-App Content, the customer and the developer have already found each other. The derived demand for services in support of the purchase of In-App Content in the In-App Aftermarket thus lacks any indirect network efforts: adding more consumers or developers does not add value to the relationship between a developer-customer pair or the associated services in support of consummating in-App transactions. Matchmaking services, critically present in the Android App Distribution Market, are not present in the In-App Aftermarket. And the Android App Distribution Market does not provide services sufficient to complete the delivery of In-App Content to the consumer. The In-App Aftermarket is therefore economically distinct from the two-sided platform market contemplated in *Ohio v. American Express* ("*Amex*"), in which the Supreme Court emphasized the determinative role of "indirect network effects,"[78] which "exist where the value of the two-sided platform to one group of participants depends on how many members of a different group participate."[79]According to the Supreme Court, when "the indirect network effects operate in only one direction," the market "behaves much like a one-sided market and should be analyzed as such."[80]

53. Moreover, I understand that Professor Schmidt will show that, as a technological matter, there is no basis for Google to insert itself into the In-App Aftermarket by requiring that developers use Google Play Billing. Google's forced insertion into the In-App Aftermarket is properly analyzed as an anticompetitive extension of the power it possesses in the separate and distinct Android App Distribution Market. My analysis first examines these two relevant markets, and my models of common impact are aimed at these markets separately, with a few caveats. To model the scenario where rival app stores compete on the dimension of a consumer subsidy (rather than on take rates), I treat the two markets as a single market, under the assumption that any enhanced Play Points could be used by consumers for both paid initial downloads and the purchase of In-App Content. I also offer a model in which rival app stores compete on the take rate assuming (in the alternative) the two markets are a single market.

III. GOOGLE'S ANTICOMPETITIVE CONDUCT IN THE PRIMARY ANDROID APP DISTRIBUTION MARKET

54. In this Part, I use data, economic methods, and evidence common to all Class members to establish that the Android App Distribution Market is a relevant market for analysis, that Google possesses monopoly power in this market, and that Google has engaged in anticompetitive conduct as a means of furthering and retaining its monopoly power in this market. Note that while I reference Google's contractual restraints impacting OEMs, mobile carriers, and

---

78. *Ohio v. American Express Co.*, 585 U.S. ___ (2018), *available at* www.supremecourt.gov/opinions/17pdf/16-1454_5h26.pdf.

79. *Id.* at 3.

80. *Id.* at 12-13 ("To be sure, it is not always necessary to consider both sides of a two-sided platform. A market should be treated as one-sided when the impacts of indirect network effects and relative pricing in that market are minor. Newspapers that sell advertisements, for example, arguably operate a two-sided platform because the value of an advertisement increases as more people read the newspaper. But in the newspaper-advertisement market, the indirect network effects operate in only one direction; newspaper readers are largely indifferent to the amount of advertising that a newspaper contains. Because of these weak indirect network effects, the market for newspaper advertising behaves much like a one-sided market and should be analyzed as such.").

developers in the context of discussing Google's power in the relevant markets, I reserve in-depth discussion of such restraints for the sections on anticompetitive conduct.

A. The Android App Distribution Market Is a Relevant Antitrust Market

55. By Google's estimate, there were over three billion Android devices in the world as of May 2021.[81] To access this customer base of Android users, developers design Apps for Android devices. Developers traditionally reach users through App stores installed on the users' device, but also can distribute their Apps to consumer devices directly from developer websites through a process known as "sideloading."[82]

56. While the Play Store is pre-installed on every Google Android mobile device, sideloading requires independent consumer knowledge of the developer's website and the circumvention of onerous technical barriers imposed by Google. *See* Part III.D.4, *infra.* As a practical consequence, the effective Android App Distribution Market consists almost entirely of Android App stores that have been pre-installed on the mobile device. Although app stores, like some other two-sided platforms, benefit from indirect network effects, the market for initial App downloads need not always tend toward monopoly.[83] Without the multiple restrictions that Google has imposed, consumers would have easier access to multiple cross-platform app stores.[84]

57. The Android App Distribution Market is a distinct relevant antitrust market under the *Horizontal Merger Guidelines*' SSNIP test. To begin, I analyze the developers' side of the market. To list their Apps, developers pay the Play Store a small up-front fee of $25 plus a take rate of revenues for paid downloads.[85] If a hypothetical app-store monopolist that distributed Android-compatible Apps were to raise its take rate above competitive levels by a small but

---

81. Alex Cranz, *There are over 3 billion active Android devices. That's a lot of smartphones*, THE VERGE, (May 18, 2021), *available at* www.theverge.com/2021/5/18/22440813/android-devices-active-number-smartphones-google-2021. The author notes this estimate is conservative because the data are "taken from the Google Play Store, which doesn't take into account devices based on Android but that use alternative stores, including Amazon Fire devices…" *Id.*

82. For the purpose of this class certification report, I exclude sideloading from the Android App Distribution Market. Sideloading is partially an artifact of Google's restraints, as some developers such as Epic pulled their App outside of the Play Store to evade Google's excessive take rates, leaving users with no choice but to sideload to access the App. *See* Nick Statt, *Apple Just Kicked Fortnite Off the App Store*, THE VERGE (Aug. 13, 2020), *available at* www.theverge.com/2020/8/13/21366438/apple-fortnite-ios-app-store-violations-epic-payments ("Epic previously bypassed Google's Play Store on Android by releasing *Fortnite* as a direct download through its own software launcher. But the studio eventually relented earlier this year after failing to appeal Google for an exemption of its similar 30 percent cut of all in-app purchases."). Given the hassles imposed on the user, many of which are imposed by Google (see Part III.D.4, *infra*), sideloading is presently an inferior substitute to downloading an App from an app store. The inclusion or exclusion of sideloading has no bearing on my opinions regarding Google's market power or on my proof of common impact.

83. *See, e.g.*, Mark Looi, *On "The Platform Delusion" by Jonathan Knee*, MEDIUM (Dec. 21, 2021), *available at* marklooi.medium.com/on-the-platform-delusion-by-jonathan-knee-a787a672b932.

84. Google has produced data calculating the "% sideloaded app" at [redacted] to [redacted] percent in 2019 and 2020. GOOG-PLAY-001508603 ("Apps by Source"). This does not represent the share of apps sideloaded by users to avoid the Play Store. The [redacted] to [redacted] percent statistic is driven by [redacted] *Id*. The same data show that less than [redacted]e percent of Apps are [redacted] *Id*.

85. Play Console Help, *How to use Play Console*, *available at* support.google.com/googleplay/android-developer/answer/6112435?hl=en#zippy=%2Cstep-pay-registration-fee ("there is a US$25 one-time registration fee").

significant amount, say by five percent in accordance with the *Guidelines*, developers on the monopoly app store would not stop distributing their Apps through that app store, because the Android App Distribution Market is too large to forgo. Android devices account for 40 percent of the mobile devices purchased in the United States, and 70 percent of mobile devices bought globally.[86] Developers cannot earn a profit until the margins on their initial downloads or related In-App Content cover the often substantial development and marketing costs.[87] As long as incremental sales of Apps are bringing value to developers and paying down their fixed costs, they would not abandon distribution through the Android platform in the event of a small but significant price increase over competitive levels. Further, sideloading is not a commercially viable alternative for most developers and therefore does not constrain Google's pricing.[88]

58. Within the Android App Distribution Market, the Play Store has power in large part because of its broad reach. Record evidence suggests that developers are attracted to the Play Store primarily because of its reach.[89] In 2015, Google asked Dr. Itamar Simonson to "conduct a survey of mobile device application developers."[90] The survey's objectives included "assessing the factors that influence developers' decisions whether to develop applications for a new or existing mobile platform."[91] Dr. Simonson concluded that the "survey findings are consistent with other studies of application developers' decision making and with Google's recognition (as reflected in internal documents) that the volume of developed applications is largely driven by the number of Android users."[92] In short, developers of Android-compatible Apps would be insensitive to a small, but significant, sustained increase in Android App distribution take rates, and hence the Android Application Distribution Market is a distinct relevant antitrust market.

59. Turning to the consumer side, while it is hypothetically possible that consumers might switch to an Apple device as a result of increased take rates by an Android app store, this hypothetical possibility is unrealistic. Consumers are not aware of take rates charged to

---

86. Statcounter, *Mobile Operating System Market Share Worldwide*, (accessed Feb. 2022), *available at* gs.statcounter.com/os-market-share/mobile/worldwide/#yearly-2019-2019-bar. Android and iOS collectively account for 99.68 percent share, while Windows accounts for 0.01 percent of the mobile operating system market worldwide. Statcounter, *Mobile Operating System Market Share United States of America,* (accessed Feb. 2022), *available at* gs.statcounter.com/os-market-share/mobile/united-states-of-america.

87. *Development Options and Costs*, Business Information Systems: Design an App for That, Table 7.1 "Various Fixed Costs", *available at* saylordotorg.github.io/text_business-information-systems-design-an-app-for-that/s11-01-development-options-and-costs.html.

88. *See, e.g.,* Koh Dep. at 50:14-51:5 (During his time at [redacted], the company did not consider offering games for direct download because it would "require too much work, engineering work."); *Id.* at 101:21-102:14 (During his time working for developers, they performed cost-benefit analyses regarding distribution via direct download, but it was "too difficult to tell" whether the user reach they would achieve would justify the investment.).

89. *Id.* at 89:25-90:9 (For developers trying to reach new users in the United States, "Google would be the priority just because it has a large volume of users."); *Id.* at 321:19-323:1 (During his time at the developer [redacted], the company decided not to distribute on other Android app platforms because "there was engineering work required for us to be able to utilize those services" and because of the size of Play's distribution platform across Android devices); *Id.* at 324:6-12 (same consideration for the developer [redacted]).

90. GOOG-PLAY-007317611 at GOOG-PLAY-007317613 (Report of Dr. Itamar Simonson, Feb. 8, 2016). Pre-testing was done at the "end of December" 2015. *Id.* at GOOG-PLAY-007317618. The survey was administered between December 28, 2015 and January 22, 2016. *Id.* at GOOG-PLAY-007317619.

91. *Id.* at GOOG-PLAY-007317613.

92. *Id.* at GOOG-PLAY-007317615.

developers.[93] Even if consumers were aware of the take rates charged to developers, they would not be aware of the Play Store's restrictions on developers nor of how such restrictions influence the prices charged to consumers for App downloads. Because Google has prohibited developers from steering consumers to lower-cost alternative platforms, consumers have little or no experience with discounting.

60. A hypothetical monopolist in the Android App Distribution Market would be able to profitably increase consumer prices above competitive levels. Even if consumers had perfect information about take rates and the restrictions on developers as well as their implications for App prices over the lifecycle of the device (which they do not have in the actual world), very few would switch their device and operating system in response to a small, but significant, difference in the take rate charged to developers by a monopolist app store. That is especially true for Android phone users due to the significantly higher prices of Apple iPhones, which are nearly three times as expensive on average as Android phones.[94] A price difference this large would require extraordinary spending by consumers on Android Apps for a five percent increase in the app store take rate to render a switch economically plausible.[95] Further, the vast majority of Android users (nearly 80 percent) keep their Android phones for over a year, and many (nearly 30 percent) keep their phones for over two years.[96] As a result of the factors contributing to lock-in explained in Part I.B, consumers display a high degree of loyalty to the operating system they have chosen and learned, and they would not switch devices due to a small, but significant, increase in the take rate or associated app price for Android App distribution.[97]

---

93. Rachel Rickard Straus, *Apple faces landmark legal claim that could pay out to millions: Rip-off that adds 30% to price of smartphone apps*, THIS IS MONEY, (Dec. 11 2021), *available at* www.thisismoney.co.uk/money/bills/article-10299235/Rip-adds-30-price-smartphone-apps.html ("Most customers do not realise they are in effect paying huge commissions. But claimants say Apple and Android users have no alternative so Apple and Google can effectively charge what they like.").

94. Amit Chowdhry, *Average iPhone Price Increases To $687 and Android Decreases To $254, Says Report*, FORBES, (Feb. 3, 2015), *available at* www.forbes.com/sites/amitchowdhry/2015/02/03/average-iphone-price-increases-to-687-and-android-decreases-to-254-says-report/?sh=d9bcd17539e4.

95. A 2021 survey found that only 5.2 percent of users chose "better prices" as a reason why Android users would consider switching to an iPhone. This implies that the switch would be unlikely and uneconomic for most, regardless of whether "better prices" meant better prices for the device or for the apps. Among the other reasons were "[l]onger software support," "Apple ecosystem integration," and "[b]etter privacy protection." *See* Abhin Mahipal, *Survey: 18% of Android users would consider switching to iPhone 13, but this is down 15% from last year*, SELLCELL, (Aug. 31, 2021), *available at* www.sellcell.com/blog/survey-18-percent-of-android-users-would-consider-switching-to-iphone-13/.

96. Consumer Intelligence Research Partners, *How Long Do Android Users Own an Android Phone*, (Sep. 21, 2016), *available at* files.constantcontact.com/150f9af2201/a238f4a1-5b70-4853-b21e-226c94104d30.pdf. It bears noting that these data are from 2016, and the trend across all devices is to keep phones for even longer periods. *See* Abigail Ng, *Smartphone users are waiting longer before upgrading — here's why*, CNBC, (May 17, 2019), *available at* www.cnbc.com/2019/05/17/smartphone-users-are-waiting-longer-before-upgrading-heres-why.html. ("In 2016, American smartphone owners used their phones for 22.7 months on average before upgrading. By 2018, that number had increased to 24.7.").

97. *See, e.g.,* Chuck Jones, *Apple's iOS Loyalty Rate Is Lower Than Google's Android, But Apple May Steal More Users Each Year*, FORBES, (Mar. 10, 2018), *available at* www.forbes.com/sites/chuckjones/2018/03/10/apples-ios-loyalty-rate-is-lower-than-googles-android-but-apple-may-steal-more-users-each-year/?sh=29b39ae68a8e ("Loyalty is also as high as we've ever seen, really from 85-90% at any given point. With only two mobile operating systems at this point, it appears *users now pick one, learn it, invest in apps and storage, and stick with it*.") (emphasis added). *See also* Consumer Intelligence Research Partners, *Mobile Operating System Loyalty: High and Steady,* (Mar. 8, 2018), *available at* files.constantcontact.com/150f9af2201/4bca9a19-a8b0-46bd-95bd-85740ff3fb5d.pdf ("CIRP

61. In sum, analyzing whether there is an Android App Distribution Market from both the developer and consumer perspectives leads to the conclusion that it is a distinct relevant antitrust market. Any competition that might exist between Google and Apple with respect to operating systems does not significantly constrain Google's ability to extract supra-competitive prices in the Android App Distribution Market. That Apple[98] (in November 2020) and Google[99] (in March 2021) lowered take rates for small developers to 15 percent within a few months of each other does not imply one take-rate reduction was caused by the other, nor does it imply that their two app stores significantly discipline each other's prices (for developers) and thus are in the same product market. The moves came just after antitrust lawsuits were filed in August 2020, and Congress published a report on the platforms' conduct in October 2020.[100] Internal Google documents also suggest that the decision to reduce its take rate on small developers was driven at least in part by public-relations and regulatory considerations.[101]

62. The availability of apps on personal computers (PCs) or consoles does not expand the relevant antitrust market. From a developer's perspective, the mobile, Android App ecosystem represents too large a customer segment to ignore.[102] Indeed, the sales of mobile games are roughly equal to the *sum* of sales of console and PC games.[103] Not only would developers sacrifice significant sales by walking away from the Play Store, but, for those who had not yet done so, they would also incur additional expense to write code for an Android-based App to work in a console or PC environment. Similarly, from a consumer's perspective, the functionality of a PC or gaming console is distinct from that of a smartphone, which explains why households commonly own all three types of devices, using each technology for distinct purposes. Further, in the case of a consumer who owns an Android phone but not a console or PC, any substitution towards consoles or PCs would require the purchase of new hardware, further reducing the viability of significant defection.

---

finds that between Android and iOS, loyalty to each has remained steady since early 2016, at the highest levels seen. Android has a 91% loyalty rate, compared to 86% for iOS, measured as the percentage of customers that remain with each operating system when activating a new phone over the twelve months ending December 2017.").

98. Kif Leswing, *Apple will cut App Store commissions by half to 15% for small app makers*, CNBC, (Nov. 18, 2020), *available at* www.cnbc.com/2020/11/18/apple-will-cut-app-store-fees-by-half-to-15percent-for-small-developers.html (describing the take-rate decrease as a "high-profile olive branch from Apple to lawmakers.").

99. Chaim Gartenberg, *Google will reduce Play Store cut to 15 percent for a developer's first $1M in annual revenue*, THE VERGE, (Mar. 16, 2021), *available at* www.theverge.com/2021/3/16/22333777/google-play-store-fee-reduction-developers-1-million-dollars.

100. *Id*. ("The new policy also comes at a critical moment when Google (and Apple's) app store policies are under intense public scrutiny, kicked off by the removal of Epic Games' *Fortnite* from both the App Store and Play Store and the game developer's subsequent antitrust lawsuits against Apple and Google.").

101. Google's documents suggest that its decision in March 2021, to reduce its take rate to 15 percent for the first $1 million in developer revenue was driven at least in part by public-relations considerations. *See* GOOG-PLAY-007317535 ("Play Store Global Research[:] All Markets Findings"); *See also* GOOG-PLAY-007317521 ("Play Store Global Research[:] India Findings"); GOOG-PLAY-007317528 ("Play Store Global Research[:] South Korea Grasstop Findings").

102. The value of transactions in the Play Store reached $38.6 billion in 2020. *See* Mansoor Iqbal, *App Revenue Data (2022)*, BUSINESS OF APPS, (Feb. 16, 2022), *available at* www.businessofapps.com/data/app-revenues/ (citing App Annie and Sensor Tower).

103. WEPC, *Console Gaming Statistics 2022,* (Jan. 20, 2022), *available at* www.wepc.com/statistics/console-gaming/.

63. As described in more detail in Part V.A.4, the Epic Games Store[104] and Microsoft[105] have both charged developers a 12 percent take rate on their respective PC game platforms. That this large disparity in take rates persists implies that developers perceive the Play Store to be a unique outlet, and developers are beholden to Google to reach their critical audience of consumers. And Google has not changed its pricing in response to changes in take rates from Microsoft, console makers, or the Epic Store.

64. Moreover, Google recognizes that cross-platform gaming is new and not expected to reach scale until the late 2020s.[106] Cross-platform gaming allows users playing the same game with different devices to play together in multiplayer modes.[107] Accordingly, cross-platform gaming has not been and is not presently a significant factor in Google's pricing to developers on its Play Store.[108]

65. "Web-based apps," or apps that reside on the Internet outside of an app store, similarly do not constrain Google's pricing on the Play Store. According to Google's documents, web-based apps are not close substitutes from the consumer's perspective to a traditional or "native" App.[109] Traditional Apps, which Google defines as being located on a user's home screen, are considered by Google to be the "preferred/guaranteed experience."[110] Some traditional Apps can support "offline mode" or can be used without an Internet connection.[111] In most cases, after a traditional App is launched, the user is returned to the place where she left,[112] another significant benefit. "Web apps," by contrast, began simply as bookmarked links to the web version of an app, opened in a browser.[113] Unlike traditional—or "native"—apps, these web apps require an internet connection,[114] perform slower than native apps,[115] cannot access useful native functions (e.g., the phone's camera),[116] and do not show up in the Android app launcher.[117] According to Google,

---

104. Epic Press Release, *The Epic Games store is now live*, (Dec. 6, 2018), *available at* www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live ("The Epic Games store is now open, featuring awesome high-quality games from other developers. Our goal is to bring you great games, and to give game developers a better deal: they receive 88% of the money you spend, versus only 70% elsewhere. This helps developers succeed and make more of the games you love.").

105. Tom Warren, *Microsoft shakes up PC gaming by reducing Windows store cut to just 12 percent*, THE VERGE, (Apr. 29, 2021), *available at* www.theverge.com/2021/4/29/22409285/microsoft-store-cut-windows-pc-games-12-percent.

106. GOOG-PLAY-000231487 at GOOG-PLAY-000231489.

107. Van Vicente, *What Does Cross-Platform Mean for Gaming and Other Apps?,* HOW-TO GEEK, (Oct. 9, 2021), *available at* www.howtogeek.com/752370/what-does-cross-platform-mean-for-gaming-and-other-apps/.

108. The opinion in *Epic v. Apple* allowed for the possibility that certain games exhibit cross-platform substitutability but concluded that these games were outliers: "However, not all games are like Minecraft or Fortnite; the market still reflects that video games are, for the most part, cabined to certain platforms that take advantage of certain features of that platform, such as graphics and processing, or mobility." Rule 52 Order After Trial On The Merits, Epic Games Inc. v. Apple, Inc., Case No. 4:20-cv-05640-YGR, at 84. Because the instant matter concerns all Apps, including non-gaming Apps, the conclusion that console and PC gaming are distant substitutes for Play Store users would be bolstered.

109. GOOG-PLAY-001882239.

110. *Id*. at GOOG-PLAY-001882256.

111. *Id*. at GOOG-PLAY-001882256.

112. *Id*. at GOOG-PLAY-001882257.

113. *Id*. at GOOG-PLAY-001882261.

114. *Id*. at GOOG-PLAY-001882264.

115. *Id*. at GOOG-PLAY-001882264.

116. *Id*. at GOOG-PLAY-001882265.

117. *Id*. at GOOG-PLAY-001882263.

even the next generation of "progressive web apps," which address some of these shortcomings, still "falls short on some native interactions, including smooth animation transitions, native gestures, native menus, [and] material UI Guidelines."[118] Web based apps are no substitute for Apps on the Play Store in the Android App Distribution Market.

B. The Relevant Geographic Android App Distribution Market Is Global (Excluding China)

66. Google's Android mobile device operating system is sold throughout the world and is installed on about three quarters of all mobile devices globally, except China, where the government restricts mobile devices and operating systems and favors Chinese providers.[119] The Play Store is installed on all Google Android mobile devices by virtue of Google's "all-or-nothing" policy with respect to its GMS suite of Apps. Given the widespread distribution of the Play Store throughout the world, developers of Android-compatible Apps, wherever they are located, have strong incentives to list their Apps on a platform that can provide for distribution worldwide. The global reach of the Play Store and the developers who seek to distribute their Apps through it thus make the geographic market for the Android App Distribution Market global. That the Apps might differ in some ways by country (for example, with different languages or different features) does not change the contours of the relevant geographic market, because those changes are not material to the economics of App distribution.

67. Application of the *Horizontal Merger Guidelines*' SSNIP test confirms that the Android App Distribution Market is broader than the United States and thus global. Suppose a not-so-hypothetical monopolist in the distribution of Android-compatible Apps, such as the Play Store in the Android App Distribution Market, were to raise its take rate by a small, but significant, sustained amount in the United States. Would it attract competition from other app stores elsewhere in the world? It is plausible that a sustained increase in the take rate for initial App downloads in the United States would attract app store entrants from other countries (ignoring Google's restriction), either independently operated stores like Aptoide that already operate in multiple countries or those launched by mobile service carriers in other countries that could readily distribute Apps in the United States. Thus, the Android App Distribution Market is worldwide and is not limited to the United States.

C. Google's Market Power in the Android App Distribution Market

68. This section begins by describing evidence that directly demonstrates Google's market power in the Android App Distribution Market. I also use indirect evidence—high market shares and entry barriers—to establish that Google has market power. My assessment of the direct and indirect evidence relies entirely on data and methods that are common to the Class.

---

118. *Id*. at GOOG-PLAY-001882274.

119. GOOG-PLAY-004253884 at GOOG-PLAY-004253894 ("Samsung Leads in All Retail Driven Markets except China") (chart shows Apple and Samsung shares in different countries, with China being dominated (60 percent) with local OEMs). It bears noting that all references to "global" markets or the use of the terms "globally" or "world" in this report assume that China is excluded.

1. Direct Evidence

69. High margins imply an ability to raise prices over competitive levels. According to Google's compilations of its profit-and-loss statement for the Play Store, excluding ads, Google earned an operating profit of [redacted] in 2019, a [redacted] percent increase over the Play Store's profit of [redacted] in 2018.[120] Google's operating profit from the Play Store, again excluding ads, jumped to [redacted] in 2020, an increase of [redacted] percent [redacted].[121] The Play Store's gross profit margin in 2020 was [redacted] percent, and its operating profit margin was [redacted] percent.[122] A separate spreadsheet shows that, in 2020 alone, Google earned an *additional* [redacted] on ads that appear in the Play Store, with almost all of those revenues falling to the bottom line.[123] In [redacted], the Play Store's gross profit margin was [redacted] percent, its operating profit margin was [redacted] percent, and its operating profit was [redacted].[124] In 2021, Google earned an *additional* [redacted] on ads that appear in the Play Store, with almost all of those revenues (again) falling to the bottom line.[125]

70. That Google profitably imposed a 30 percent take rate on developers for most paid App downloads is direct evidence of its market power over developers in the Android App Distribution Market. As shown in Part V.B.4, *infra*, this take rate is high relative to competitive benchmarks. In the presence of competition, developers would be able to offer their Apps on Android devices through multiple app stores; a developer unwilling to pay a 30-percent take rate could choose to market and distribute its App on a competing app store without losing access to most customers.

71. Google's dominance in the Android App Distribution Market, reflected in its high profits and excessive take rate, flows from its power in the licensed mobile device operating systems market. Indeed, the Android App Distribution Market could be characterized as an aftermarket to the market for licensed mobile device operating systems. Google's documents illustrate how Google's power in the market for licensed mobile device operating systems helps to ensure the Play Store's dominance. A 2019 presentation reviewing the Play Store's business model displays "[t]he 50,000-foot view,"[126] showing Google Android's reach as the first in a chain of factors allowing the Play Store to be monetized.

---

120. GOOG-PLAY-000416245.
121. *Id.*
122. *Id.*
123. GOOG-PLAY-001090227 (showing Play advertising revenue of [redacted] in 2020, with cost of sales of just [redacted], direct operating expenses of [redacted], and cost allocations of [redacted]). These data imply that, in 2020, Google earned a gross margin on Play Store advertising revenue of over [redacted] percent (equal to [([redacted]). Similarly, the 2020 operating margin inclusive of direct costs is [redacted] percent (equal to [([redacted]), and the 2020 operating margin inclusive of direct costs and cost allocations is [redacted] percent (equal to [([redacted]).
124. GOOG-PLAY-010801682 (showing 2021 Play Store revenue (excluding ads) of [redacted], and gross profit of [redacted], for gross profit margin of [redacted] percent. The Play Store's operating profit was [redacted], for an operating profit margin of [redacted] percent).
125. GOOG-PLAY-010801680 (showing Play Store advertising revenue of [redacted] in 2021, with a gross profit of [redacted], for a gross profit margin of [redacted] percent. Total operating expenses (inclusive of direct costs and cost allocations) are recorded at [redacted], yielding an operating profit of [redacted] for an operating margin of [redacted]
126. GOOG-PLAY-000443763 at GOOG-PLAY-000443768.

FIGURE 2: INTERNAL GOOGLE VIEW ON LINKAGE BETWEEN
THE PLAY STORE'S DOMINANCE AND ANDROID'S DOMINANCE

Source: GOOG-PLAY-000443763 at GOOG-PLAY-000443768.

The presentation recognizes that Google's Mobile Application Distribution Agreements ("MADAs") with OEMs are used to "[t]rade access to Google's apps for placement;"[127] in other words, Google has leveraged its GMS suite of apps and APIs to ensure that the Play Store is available and prominently displayed to the user.

72. Google considered adjustments to its business model and to its take-rate levels, including "progressive rev-share rates," in which Google would "change the fundamental Play biz model for dev[elopers]," by allowing the take rate to fall over time.[128] The pressure to reduce take rates would come from OEMs capable of "building significant distribution scale" and from game developers with "highly monetizing titles," presenting an "increasing risk of going-it-alone."[129] Google observed that the obvious downside of reducing the take rate is that it would decrease profit margins,[130] and questioned whether any reduction should be done at all.[131] Since the (undated) presentation, which refers to 2018 Search and Play revenues,[132] however, Google has done little to lower take rates (outside of rates for subscriptions and small developers).

73. Google's multiple restraints affecting the Android App Distribution Market—on OEMs, carriers, and developers—coupled with its technical barriers protect Google's market

127. *Id*. at GOOG-PLAY-000443769.
128. *Id*. at GOOG-PLAY-000443775.
129. *Id*.
130. *Id*. at GOOG-PLAY-000443775-776.
131. *Id*. at GOOG-PLAY-000443776-777.
132. *Id*. at GOOG-PLAY-000443770.

power in the Android App Distribution Market. As just one of the many examples of the impact of these restraints, Google's all-or-nothing bundling of the GMS suite gave it a large competitive advantage as developers were inclined to offer Apps on a platform that was guaranteed to also carry the high-value GMS Apps.[133] Google's Jim Kolotorous testified that he was unaware of any smart phone launched outside of China without installing a single Google App or service on the device.[134]

2. Indirect Evidence

74. Google's power in the Android App Distribution Market can also be gleaned indirectly via its high shares and entry barriers. As of October 2021, a total of over 21.6 billion Apps had been downloaded from the Play Store.[135] Excluding China, where the Play Store is blocked,[136] "Apple and Google control more than 95 percent of the app store market share through iOS and Android…The app economy was built on these two platforms[.]"[137] Due in part to the massive installed base of mobile Android devices, significantly more apps are downloaded from the Play Store than from the Apple App Store. For example, in 2020, there were approximately 108.5 billion downloads from the Play Store, compared with 34.4 billion through Apple App Store.[138] Excluding China, the Play Store accounted for over 80 percent of the combined global downloads from the Play Store and the Apple App Store in 2020.[139]

---

133. Benjamin Edelman & Damien Geradin, *Android and competition law: exploring and assessing Google's practices in mobile*, 12(12-3) EUROPEAN COMPETITION JOURNAL 159-194, 162-163 (2016) [hereafter Edelman & Geradin].

134. Deposition of Jim Kolotorous at 110:13-111:14. Amazon, one of Mr. Kolotorous's exclusions, only produces tablets, not smartphones. *Id.* 125:14-17.

135. Terry Stancheva, *17 App Revenue Statistics – Mobile Is Changing the Game in 2021*, TECHJURY, (Feb. 6, 2022), *available at* techjury.net/blog/app-revenue-statistics/#gref (citing Statista).

136. Sherisse Pham, *Google now has two apps in China, but search remains off limits*, CNN BUSINESS, (May 31, 2018), *available at* money.cnn.com/2018/05/31/technology/google-in-china-files-app/index.html ("The company's own app store, Google Play, remains blocked in China[.]").

137. David Curry, *App Store Data (2022),* BUSINESS OF APPS, (Jan. 11, 2022), *available at* www.businessofapps.com/data/app-stores/

138. David Curry, *App Data Report,* BUSINESS OF APPS (2022) at 16-17 (showing 34.4 billion downloads via iOS and 108.5 billion via Google Play).

139. According to Sensor Tower, The Play Store's total global downloads in 2020 (excluding China) were 108.759 billion (6.3 billion + 22.5 billion + 5.7 billion + 2.8 billion + 21.1 billion + 49.9 billion + 459 million = 108.759 billion). *See* Sensor Tower, *2021 – 2025 Mobile Market Forecast* (2021) at 15. The Apple App Store's total global downloads in 2020 were 34.297 billion (8.9 billion + 2.0 billion + 312 million + 1.3 billion + 7.3 billion + 13.9 billion + 585 million = 34.297 billion). *Id* at 14. 8.2 billion of these downloads come from China. *Id* at 31. Therefore, in 2020, the Play Store accounted for over 80.6 percent combined global downloads from the Play Store and the Apple App Store (equal to 108.759 billion / (108.759 billion + 34.297 billion – 8.2 billion)). Note that consumer expenditures in the Apple App Store exceed those in the Play Store, despite the fact that far more apps are downloaded through the Play Store. *See, e.g.,* David Curry, *App Data Report,* BUSINESS OF APPS (2022) at 29 ("Even though Google Play has a larger installed base and 75% of all apps are downloaded on the platform, Apple's App Store leads the way in revenue."). Download statistics differ from consumer expenditure statistics in at least two important ways. *First*, many apps are downloaded free of charge. *Second*, industry data on consumer expenditures aggregates consumer expenditures on initial downloads with consumer expenditures on in-app purchases. *See, e.g., Id*. at 30 (showing aggregate "iOS App and Game Revenues" of $72 billion in 2020); *see also* Sensor Tower, *2021 – 2025 Mobile Market Forecast* (2021) at 36 (also showing a total of $72 billion in 2020 for "App Store Spending" on "Apps" and "Games"; explaining that "[d]riven by the significant success of the *in-app subscription model*, app revenue increased 4.7x between 2016 and 2020, compared to an increase of 2x for games.") (emphasis added). Note also that, based on the

75. Data from industry analysts on mobile app expenditures (which aggregates consumer expenditures on both initial downloads and in-App purchases) confirm (1) that the Play Store and the Apple App Store account for the vast majority of mobile app expenditures outside China; and (2) that the Play Store alone accounts for the vast majority of mobile app expenditures outside China and distinct from iOS. In light of the global dominance of the Play Store and the Apple App Store, industry reports covering mobile apps often focus almost exclusively on these two platforms.[140] But even when other platforms are considered, the data confirm that the Play Store and the Apple App Store account for the vast majority of mobile app expenditures outside of China. For example, global consumer expenditures in Apple's App Store outside of China in 2018 were $32.9 billion,[141] while global consumer expenditures in the Play Store (all of which is outside of China) came to $25 billion in 2018.[142] Global mobile app expenditures outside of China in 2018 were $62 billion.[143] Therefore, outside of China, the Apple App Store and the Play Store accounted for 93.4 percent of global mobile app expenditures as of 2018 (equal to [$32.9 billion + $25.0 billion]/[$62 billion]). Using the same statistics, the Play Store accounted for 85.9 percent of non-Apple mobile app expenditures outside of China in 2018 (equal to [$25.0 billion]/[$62 billion - $32.9 billion]).

76. In 2020, global consumer expenditures in Apple's App Store outside of China came to $52 billion,[144] while global consumer expenditures in the Play Store (all of which is outside of China) came to $39 billion in 2020.[145] Global mobile app expenditures outside of China in 2020 were $95 billion.[146] Therefore, outside of China, the Apple App Store and the Play Store accounted for 95.8 percent of global mobile app expenditures as of 2020 (equal to [$52 billion + $39 billion]/[$95 billion]). Using the same statistics, the Play Store alone accounted for 90.7 percent of non-Apple mobile app expenditures outside China in 2020 (equal to [$39 billion]/[$95 billion - $52 billion]).

77. Similarly, statistics on downloads (as opposed to consumer expenditure) indicate that the Play Store's share of the non-China Android App Distribution Market can be estimated at

---

Play Store's financials, the vast majority of Play Store revenue comes from [redacted], as opposed to [redacted]. *See, e.g.*, GOOG-PLAY-010801682 (showing [redacted] percent of 2021 Play Store revenue coming from [redacted]).

140. *See, e.g.,* Sensor Tower, *Global Consumer Spending in Mobile Apps Reached $133 Billion in 2021, Up Nearly 20% from 2020,* (Dec. 2021), *available at* sensortower.com/blog/app-revenue-and-downloads-2021 (reporting "Global Consumer Spending in Mobile Apps and Games" as the sum of Google Play and the App Store). *See also* David Curry, *App Data Report,* BUSINESS OF APPS (2022); *see also* Sensor Tower, *2019 – 2023 Mobile Market Forecast,* (2019).

141. In 2018, global consumer expenditures reached $47 billion in the Apple App Store. *See* Sensor Tower, *2019 – 2023 Mobile Market Forecast,* (2019) at 4. Consumer expenditures in the Apple App Store in China were $14.1 billion in 2018. *Id.* at 15. Therefore, non-China Apple App Store expenditures in 2018 were $47 billion - $14.1 billion = $32.9 billion.

142. *Id.* at 4.

143. *See* David Curry, *App Data Report,* BUSINESS OF APPS, at 44 (2022) (showing 2018 non-China revenue of $20 billion (United States) + $15 billion (Japan) + $11 billion (Europe) + $16 billion (Rest of World) = $62 billion).

144. In 2020, global consumer expenditures reached $72 billion in the Apple App Store. *See* Sensor Tower, *2021 – 2025 Mobile Market Forecast,* (2021) at 6. Consumer expenditures in the Apple App Store in China were $20 billion in 2020. *Id.* at 22. Therefore, non-China Apple App Store expenditures in 2020 were $72 billion - $20 billion = $52 billion.

145. *Id.* at 6.

146. *See* David Curry, *App Data Report,* BUSINESS OF APPS, 1-69, 44 (2022) (showing 2020 non-China revenue of $32 billion (United States) + $20 billion (Japan) + $14 billion (Europe) + $29 billion (Rest of World) = $95 billion).

over 90 percent:[147] Nearly 60 percent of non-iOS apps downloaded worldwide in 2020 were downloaded through the Play Store,[148] but this estimate vastly understates the Play Store's share of Android downloads, as the denominator includes downloads in China, where the Play Store is blocked. Removing non-iOS downloads in China would likely place the Play Store at over 90 percent of the non-China Android App Distribution Market. For example, if iOS downloads account for one quarter of all mobile downloads in China,[149] then the Play Store's share of the non-China Android App Distribution Market would be 97 percent.[150] Internal Google documents show that, on Android devices outside of China, the Play Store accounts for the vast majority of monthly app store visits and time spent in app stores.[151]

78. With respect to entry barriers, as a two-sided platform, the Play Store benefits from indirect network effects, which serve to entrench its market power;[152] additional Apps attract users, which in turn attract developers, a virtuous cycle that rewards first movers and thwarts later potential entrants. Google recognizes the power that it gets from these network effects.[153] In addition, Google Android's significant share of the number of all mobile devices,[154] means that developers have strong incentives to make their apps Android-compatible to cover the fixed costs of app development. Thus, developers effectively must list their Apps on the Play Store and agree to its restrictive conditions, including the prohibition on steering users to rival app stores. These

---

147. As explained above, download statistics differ from consumer expenditure statistics in at least two important ways. *First*, many apps are downloaded free of charge. *Second*, as explained above, industry data on consumer expenditures aggregates consumer expenditures on initial downloads with consumer expenditures on in-app purchases.

148. According to Statista, the total number of app downloads in 2020 was 218 billion, with the Play Store and iOS accounting for 143.2 billion. Statista, *Number of mobile apps downloaded worldwide from 2016 to 2020*, *available at* www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/; Statista, *Combined global Apple App Store and Google Play Store app downloads from 1st quarter 2015 to 4th quarter 2021*, *available at* www.statista.com/statistics/604343/number-of-apple-app-store-and-google-play-app-downloads-worldwide/. The Play Store was responsible for 108.5 billion downloads in 2020 and iOS had 34.4 billion, for a total of 142.9 billion. *See* Mansoor Iqbal, *App Download Data (2022),* BUSINESS OF APPS, (Jan. 11, 2022), *available at* www.businessofapps.com/data/app-statistics/ [hereafter *App Download Stats*]. This implies that the Play Store was responsible for 59 percent of the non-iOS downloads worldwide. In 2020, there were 183.6 billion non-iOS downloads (218 billion less 34.4 billion) and 108.5 billion Play Store downloads. Dividing 108.5 by 183.6 yields 0.59.

149. *See* Statista*, Market share of mobile operating systems in China from January 2013 to December 2021**, *available at* www.statista.com/statistics/262176/market-share-held-by-mobile-operating-systems-in-china/ (In October, November, and December of 2021, iOS accounted for 18.99, 19.28, and 21.08 percent of mobile operating systems in China, respectively. In the last quarter of 2021, Apple accounted for approximately (18.99 + 19.28 + 21.08)/3 = 19.8% of mobile operating systems in China.) I conservatively set the percentage to 25 percent for purposes of my calculations here.

150. There were 96.2 billion downloads in China in 2020. *See App Download Stats*, *supra*. Assuming one quarter of those were iOS, then non-iOS downloads in China would be 72.15 billion (equal to 96.2 billion * (1 – 1/4)). Therefore, total worldwide non-iOS downloads outside China would amount to 183.6 billion – 72.15 billion = 111.45 billion. The Play Store share of the non-China Android App Distribution Market would then equal 108.5 billion / 111.45 billion = 0.97.

151. *See* GOOG-PLAY-002076224.R (Google 2019 slide deck titled "OEM App Store Share Analysis", summarizing global analysis of Android app store visits (excluding China) as follows: "For each region, the majority (>90%) of all app store visits in a month are to Play Store.") *Id.* at slide 5. Google's analysis also shows that the Play Store accounts for approximately 95 percent of monthly app store time spent of Android app stores worldwide excluding China. *See* GOOG-PLAY-002076224.R at GOOG-PLAY-002076236.R.

152. *See generally* Jean-Charles Rochet & Jean Tirole, *Platform Competition in Two-Sided Markets*, 1(4) EUROPEAN ECONOMIC ASSOCIATION 990-1029 (2003) [hereafter Rochet & Tirole].

153. GOOG-PLAY-000879194.R ("Amazon competitor deep dive" April 2017).

154. *Statista Mobile OS Shares*, *supra*.

restrictions therefore act as substantial barriers to entry for effective competition from rival app stores.

79. The impact of indirect network effects in the Android App Distribution Market is reflected in these statistics: the Play Store offered 3.5 million Apps by the first quarter of 2021,[155] the most of any app store in the Android App Distribution Market. In comparison, the Amazon Appstore offered approximately 460,000 Android Apps.[156] Google recognizes the power of network effects in giving its store an advantage: "Play benefits from network effects. Users come to Play because we have by far the most compelling catalogue of apps/games. Developers come to Play because that's where the users are."[157] Google also admits significant impediments to competition, noting that the process of sideloading the Amazon Appstore involves 14 steps, is "quite complex," and the hurdle is "too high for most users" to take advantage of any discounts offered by Amazon.[158]

80. As discussed more fully in Part III.D.3 below, Google's conduct vis-à-vis developers—including preventing developers from steering users to rival stores and conditioning developers' access to valuable advertising programs on YouTube and Google Search on the sale and distribution of developers' Apps in the Play Store—has substantially foreclosed potential opportunities for alternative app stores to compete with Google, effectively hindering their ability to develop into viable alternative distribution channels for developers. Absent these provisions, developers would have been more inclined to participate in and promote (via steering) alternative app stores, such as those owned by Amazon or LG. Nor has Samsung's Galaxy Store provided effective competition for the Play Store. Although many Android devices come preloaded with both the Play Store and Samsung's Galaxy Store,[159] as explained more fully in Part III.D.2.d below, in part due to Google's conduct, the Galaxy Store has not gained widespread traction with developers;[160] it appears only on Samsung devices, and is thus more limited in reach than the Play Store and which serves as a deterrent to some developers from investing in the Galaxy Store.[161]

81. The data also show that non-Galaxy app stores achieved only trivial penetration, as they have been pre-loaded on fewer than eleven percent of all devices. As seen below in Table 1, between 2018 and 2021, the [redacted] was on fewer than [redacted] percent of active devices, and the [redacted] was on fewer than [redacted] percent of active devices.[162]

155. *See* Statista, *Number of apps available in leading app stores as of 2021*, *available at* www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/.

156. *Id.*

157. GOOG-PLAY-000879194.R at GOOG-PLAY-000879207.R.

158. *Id.* at GOOG-PLAY-000879204.R.

159. Data from GOOG-PLAY-007203253. *See* Table 1, *infra*.

160. *See, e.g.,* Daria Dubrova, *9 Alternative Android App Stores*, THE APP SOLUTIONS, *available at* theappsolutions.com/blog/marketing/alternative-android-app-stores/ ("From the company that manufacturers the most Android phones it is no surprise that Samsung has developed their own app store. Compared to other app stores, Samsung Galaxy Apps has a relatively small number of apps but this can be positive for apps to stand out.").

161. Koh Dep. at 323:3-23 (Nexon didn't invest in the Galaxy Store given its limited reach).

162. GOOG-PLAY-007203253.

TABLE 1: SHARE OF ACTIVE ANDROID DEVICES WITH ALTERNATIVE APP STORES

Year	[redacted]	[redacted]	[redacted]
2018	1.47%	8.3%	52%
2019	0.83%	9.5%	55%
2020	0.47%	9.7%	57%
2021	0.33%	8.7%	57%

Source: GOOG-PLAY-007203253

The economic literature recognizes that evidence of high market shares combined with barriers to entry strongly imply Google's market power in the Android App Distribution Market was sufficient to raise prices above the competitive level.[163] Such evidence is clearly present here.

D. Google's Exclusionary Conduct in the Android App Distribution Market

82. Google's use of various restraints to maintain its dominance in the Android App Distribution Market inhibits competition from rival app stores on mobile devices and from sideloading of direct downloads from developers' websites. Such competition would enable consumers and developers to readily connect to more than one competitive platform, a practice known as "multi-homing." A developer can take advantage of multi-homing by discounting the price of its Apps to "steer" consumers to use the lower-cost platform. Steering and multi-homing combined generally would lower the equilibrium take rate charged by each platform. While Google has claimed openness to other app stores,[164] I next discuss how it has effectively utilized contractual restrictions and revenue-sharing agreements with mobile carriers, OEMs, and developers to restrain competition.[165] I also understand that Professor Schmidt finds that Google imposes overly broad technological barriers that inhibit the installation and usage of alternative app stores on Google Android devices, compared with the Play Store.

---

163. Large entry barriers tend to eliminate the possibility a competitive fringe can readily and substantially increase production in response to a small increase in the incumbent's price. *See, e.g.,* William Landes & Richard Posner, *Market Power in Antitrust Cases,* 94(5) HARVARD LAW REVIEW 937-996, 947 (1981) [hereafter Landes & Posner].

164. *See, e.g.*, Sameer Samat, *Listening to Developer Feedback to Improve Google Play,* ANDROID DEVELOPERS BLOG, (Sep. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html ("We believe that developers should have a choice in how they distribute their apps and that stores should compete for the consumer's and the developer's business. Choice has always been a core tenet of Android, and it's why consumers have always had control over which apps they use, be it their keyboard, messaging app, phone dialer or app store. Android has always allowed people to get apps from multiple app stores. In fact, most Android devices ship with at least two app stores preinstalled, and consumers are able to install additional app stores. Each store is able to decide its own business model and consumer features. This openness means that even if a developer and Google do not agree on business terms the developer can still distribute on the Android platform. This is why Fortnite, for example, is available directly from Epic's store or from other app stores including Samsung's Galaxy App Store.").

165. *See, e.g.*, GOOG-PLAY-003776161.R at GOOG-PLAY-003776176.R (table on common partner types and what they do, including mobile carriers that "maintain Search exclusivity on devices sold through their channel," and OEMs that "pre-install suite of Google Apps; Google Search exclusivity.").

1. Google's Revenue-Sharing Agreements Eliminated the Threat of Competition from Mobile Carriers

83. Commencing with Google's rollout of the Android operating system, Google prevented competition in the Android universe—and in the creation of rival app stores—through revenue-sharing agreements ("RSAs") with major mobile carriers. Google identified mobile carriers as the most significant initial threat to Google's control of the Android ecosystem because they had the ability to customize the mobile devices they shipped. In 2009, Google recognized that "the greatest threat to Android Market[place] is that carriers can easily set up their own, controlled, application market that will be the default on devices that are not co-developed by Google."[166]

84. Google therefore implemented revenue sharing from the Play Store and other Google properties to neutralize that threat. Google sought to neutralize that threat by "giving generous revenue share that more or less matches what they would make from their own markets," until a point where "carriers will be unable to compete with their own offerings because their own offerings will be so limited in comparison."[167] Indeed "Rev share is a big (if not 'the') reason that carriers are loosening their grip on app distribution."[168] By 2010, Google knew that "carrier support is critical to gain traction against Apple, and to gain consumer support over incumbent and emerging platforms."[169] Record evidence indicates that Google was aware that most carriers to which it offered revenue share deals were pursuing their own stores, and were therefore potential competitors with Google's emerging app store.[170] Google anticipated that its revenue sharing with carriers would incentivize them to promote Google's store over their own stores.[171] Although the agreements technically allowed carriers to continue pursuit of their own stores, record evidence indicates that the practical effect was to focus the carriers elsewhere—specifically, on the Play Store.[172] Payments to potential competitors not to enter, referred to as "pay-for-delay" agreements in the pharmaceutical industry, are recognized as being anticompetitive under certain conditions.[173] Another potential economic lens with which to assess these payments is a form of predation, which I discuss below.

85. [redacted] In 2009, as part of a broader agreement, Google entered into a revenue-sharing agreement with Verizon, wherein Google agreed to provide Verizon with 25 percent of each App

---

166. GOOG-PLAY-001423609.
167. *Id.*
168. GOOG-PLAY-001381141.
169. GOOG-PLAY-004541676 at GOOG-PLAY-004541679.
170. *See* Rosenberg Dep. at 174:3-181:14; GOOG-PLAY4-000339939; GOOG-PLAY-001381054; GOOG-PLAY-001423609.
171. GOOG-PLAY4-000339939; GOOG-PLAY-001423609.
172. *Id.*; *see also* GOOG-PLAY-001423609; GOOG-PLAY-008427238.
173. *See, e.g.*, Kevin Caves and Hal Singer, *On the Utility of Surrogates for Rule of Reason Cases*, CPI ANTITRUST CHRONICLE, (2015); Aaron Edlin, Scott Hemphill, Herb Hovenkamp, and Carl Shapiro, *Activating Actavis*, ANTITRUST, (2013) (explaining how the likelihood that a payment from a branded drug to a generic is anticompetitive increases when the payment exceeds the avoided litigation costs or value of services rendered).
174. GOOG-PLAY-001400503; GOOG-PLAY4-000284361; [redacted]

transaction, while Google retained only five percent.[175] Google continued to pay Verizon a 25 percent share of app credit card sales through June 2014.[176] [redacted] By 2014, [redacted], Google had reached Play Store RSAs on similar terms with carriers around the world.[178]

86. During 2009 to 2012,[179] when Google was retaining up to five percent of the developers' revenues—and ceding the residual 25 to 27 percent of developer revenues to OEMs and mobile carriers—Google was not covering its marginal costs, and thus not covering its average variable costs,[180] of operating the Play Store. According to its own financial data, Google's gross profit from the Play Store was negative as late as 2011, and operating profit was negative into 2012.[181] Google's documents indicate that paying carriers 25 percent for distribution of the Play Store would have resulted in losses for Google, making prices below average variable costs on every App transaction involving payment. For instance, an October 2013 presentation indicates that Google's first full year of positive gross margins for the Play Store was 2013 and, critically, the gross margin of $0.13 per $1 in consumer spend would have approached zero had Google continued to pay $0.25 per dollar to carriers rather than $0.14 per dollar (a difference of $0.11).

---

175. GOOG-PLAY-001400503 at GOOG-PLAY-001400530 (§ 14.12(b)); GOOG-PLAY4-000284361 at GOOG-PLAY4-000284365.

176. GOOG-PLAY-004542110; GOOG-PLAY-000131205.R at GOOG-PLAY-000131232.R.

177. [redacted]. When Direct Carrier Billing was involved, the carriers received 27 percent of revenue, leaving even less for Google. *See* GOOG-PLAY-004499366 at GOOG-PLAY-004499369 (noting that "Margins increasing due to (i) Renegotiated carrier deals from 27% on DCB and 25% on Credit Cards on Apps to 15% and 0% respectively . . .").

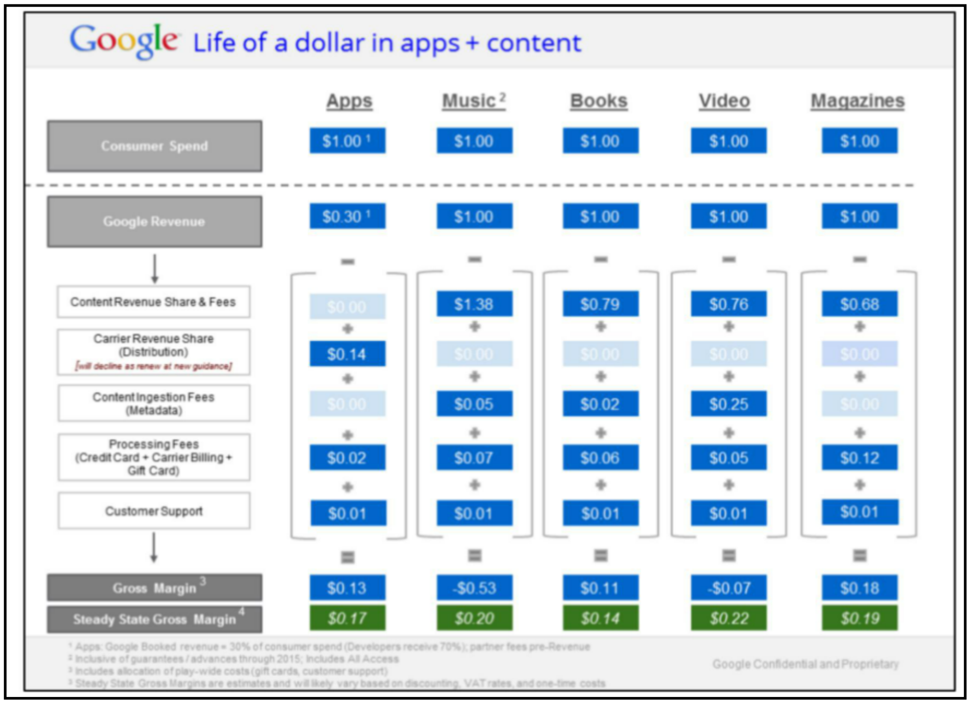
178. *See, e.g.* GOOG-PLAY-001886008.R at GOOG-PLAY-001886011.R (showing revised app revenue share percentages for various carriers in 2013, with "previous rates" as 25% credit card, 27% direct carrier billing ("DCB") and "new guidance" as 0% credit card, 15% DCB); GOOG-PLAY-000131205.R at GOOG-PLAY-000131232.R-GOOG-PLAY-000131233.R (2014 presentation showing app revenue share percentages for various carriers at 25% credit card, 25%-27% DCB); GOOG-PLAY-001385324 at GOOG-PLAY-001385345 (2010 presentation noting that "rev share % paid to developers and carriers" is 95 percent total).

179. *See* Part III.D.1.

180. Average variable cost refers to variable costs divided by the *total* quantity of output produced. Marginal cost refers to the additional cost incurred by producing *one additional unit* of output. There may be certain variable costs such as customer support that Google incurs when output expands considerably, but that are not incurred when output expands by one unit. Thus, Google's average variable cost will always exceed its marginal cost. Because Google continues to incur expenses at the margin such as processing fees for each sale, its marginal costs are a good approximation of its average variable costs.

181. GOOG-PLAY-000416245 (showing gross profit of negative $5 million in and operating profit of negative $126 million in 2011. Gross profit is positive $75 million in 2012; operating profit is negative $72 million in 2012).

FIGURE 3: GOOGLE'S GROSS MARGINS ON THE PLAY STORE IN 2013



	Apps	Music [2]	Books	Video	Magazines
Consumer Spend	$1.00 [1]	$1.00	$1.00	$1.00	$1.00
Google Revenue	$0.30 [1]	$1.00	$1.00	$1.00	$1.00
Content Revenue Share & Fees	$0.00	$1.38	$0.79	$0.76	$0.68
Carrier Revenue Share (Distribution) [will decline as renew at new guidance]	$0.14	$0.00	$0.00	$0.00	$0.00
Content Ingestion Fees (Metadata)	$0.00	$0.05	$0.02	$0.25	$0.00
Processing Fees (Credit Card + Carrier Billing + Gift Card)	$0.02	$0.07	$0.06	$0.05	$0.12
Customer Support	$0.01	$0.01	$0.01	$0.01	$0.01
Gross Margin [3]	$0.13	-$0.53	$0.11	-$0.07	$0.18
Steady State Gross Margin [4]	$0.17	$0.20	$0.14	$0.22	$0.19

These 2013 figures indicate that anything approaching a 25 percent payment to the carriers would not have been a sustainable long-term strategy, as it would have been below Google's marginal costs and hence below its average variable costs.[182]

87. Based on its financial data, my best estimate is that Google's marginal costs during the period 2016 through 2020 are approximately [redacted] to [redacted] percent of revenue, which exceeds its [redacted] percent revenue share net of payments to U.S. mobile carriers.[183] Accordingly, an equally efficient rival app store that sought to compete with Google by mimicking its payments to OEMs and mobile carriers would not be able to operate profitably.[184] Google was able to sustain such losses via other revenues (such as advertising revenues from Search), and as demonstrated below, it eventually anticipated recouping any losses from the early days of the Play Store, after the threat of competitive entry subsided, by curtailing payments to OEMs and mobile operators.

88. According to Tom Moss, Google's Head of Japan new business development, Google perceived the Android Market as the "bitter pill" for mobile carriers, and the revenue share

182. GOOG-PLAY-004499366 at GOOG-PLAY-004499378.

183. *See* Table 5, *infra* (showing In-App Aftermarket Impact & Damages (8/16/2016 – 12/31/2020). Row 7 calculates marginal cost at [redacted] percent of consumer expenditures. As explained below, this includes all direct costs of sales and direct operating expenses. If one considers transaction fees in isolation, Google's financial data for the Play Store show that transaction fees alone account for approximately [redacted] percent of consumer expenditures. *See* Part V.C.1, *infra*. *See also* GOOG-PLAY-000416245.

184. I understand from Plaintiffs' counsel that, as a legal matter, predation does not require pricing to customers below costs, but also can be established via demonstration that payments to *input providers* do not permit rivals to survive. *See* Weyerhaeuser Co. v. Ross-Simmons Hardwood Lumber Co., 549 U.S. 312 (2007) ("A predatory-bidding plaintiff must prove that the predator's bidding on the buy side caused the cost of the relevant output to rise above the revenues generated in the sale of those outputs."). Here, the cost of the relevant output (the cost of operating the Play Store) was approximately eight percent of Google's revenues, which exceeded the revenues generated in the sale of those outputs (five percent) by three percentage points.

as the "sugar" that makes the pill go down.[185] In 2009, Google's plan was to make the initial offer sweet enough for the carriers so that they would not distribute or invest in their own app stores on their phones. Then there would be a "tipping point" when all of the users and developers were wedded to the Play Store, when Google could keep more of the developer revenue for itself.[186] By that point, Google would be able to taper down the payments to mobile operators, which is precisely what it did. Google began reducing the payments to carriers in 2013, and it eventually kept the full 30 percent of developer revenues,[187] easily covering its marginal costs (of approximately [redacted] percent of revenues) and earning an operating profit of [redacted] by 2018.[188] By the time the Android App Distribution Market had "tipped" in Google's favor, there was very little chance that a mobile operator would enter with its own app store. Not only did Google plan to recoup its losses from its predatory payments to mobile carriers, it executed its recoupment strategy to perfection.[189]

89. Critically important to Google's predatory strategy was to change the rules in the middle of the game, but only after developers were dependent on Google's ecosystem. Google perceived its strategy as building in multiple "phases."[190] The first phase was "ecosystem building," which spanned 2008 through 2010, and entailed having "multiple OEMs/devices" and "multiple carriers" distribute Android.[191] The second phase was to "extend Google's core business," which entailed extending its power from search and the associated advertisements into adjacent markets such as app distribution. In 2013, Google would move into the final phase, titled "Change the rules/Get a better deal."[192] After widespread adoption of Google's Android ecosystem and the Play Store, Google could "increase Google net rev. on apps."[193] In sum, Google's anticompetitive strategy for securing the Play Store's dominance entailed changing the rules after millions of users, multiple carriers, and multiple OEMs had adopted Google's ecosystem.

90. As Google's market power grew, Google reduced the carrier's revenue shares (as well as the OEM's revenue shares abroad). In a 2014 Google strategy document, the question of "what would happen if we ceased paying rev. share to carriers?" was answered: "[T]hey will look at other monetization alternatives for search and apps. If we do it broadly enough competing solutions will take full advantage of it."[194]

185. GOOG-PLAY-001423609.

186. *Id.* ("The solution, of course, is to create consumer demand for Android Market so that the carriers have no choice but to install it and make it easily available. How do we accomplish that? By giving generous revenue share that more or less matches what they would make from their own [app] markets.").

187. GOOG-PLAY-000443763 at GOOG-PLAY-000443772.

188. GOOG-PLAY-000416245.

189. Typically, predation is used to denote charging prices below costs. But the term may also refer to a situation when an input provider (here, the carrier or OEM) is paid so much that an equally efficient rival would not find it profitable to match the payment to the input provider while competing, in which case the payments may drive out equally efficient rivals.

190. GOOG-PLAY-001337211 ("Android: OC Quarterly Review – Q4 2010, Oct 12, 2010") at GOOG-PLAY-001337226.

191. *Id.*

192. *Id.*

193. *Id.*

194. GOOG-PLAY-007264058 at GOOG-PLAY-007264062.

91. [redacted], in 2013 and 2014, Google only paid AT&T five percent revenue share for App credit card sales, keeping 25 percent.[198] [redacted] And Google finally bested Verizon, successfully amending their revenue-sharing agreement in late 2014 to reduce App credit card sales revenue-share payments to ten percent for nine months, five percent for one year thereafter, and then zero after June 2016.[200] A 2015 internal Google document lauded the multi-million-dollar reduction in the Play Store's cost of sales because Google had "re-negotiated deals with the major carriers at the end of last year which led to significant savings."[201]

92. The importance of the carrier channel is evident in Google's internal documents. Even in 2019, Google still expressed concern that [redacted] could lead to mobile devices being sold with "[redacted]" and that, if "[redacted]"[202] Google feared that "[c]arriers would configure Android devices in a way most profitable to them," through these "alternative app stores."[203]

93. As explained in Part III.C.2, because the (two-sided) Android App Distribution Market is uniquely characterized by strong indirect network effects, the barriers to entry for a rival app store made for a dangerous probability of recoupment of its early losses by Google. Google's control over mobile devices meant that developers had strong incentives to make their apps Android-compatible; additional apps attracted users, which in turn attracted developers—a virtuous cycle that entrants could not exploit, at least in part due to Google's strategy. Indeed, the probability of recoupment was so high that we observe no significant competition in the Android

---

195. [redacted]

196. [redacted]

197. [redacted]

198. GOOG-PLAY-003604606 at GOOG-PLAY-003604619. AT&T collected 9 percent net GTV for direct-carrier billing transactions.

199. GOOG-PLAY-003604601 at GOOG-PLAY-003604603. [redacted]

200. GOOG-PLAY-003605103; GOOG-PLAY4-002178046 at GOOG-PLAY4-002178049; GOOG-PLAY-002891881 at GOOG-PLAY-002891882 (2016 internal Google email noting Verizon's complaint that "dcb payments are insignificant and they want the cc back again").

201. GOOG-PLAY4-004677224 at GOOG-PLAY4-004677225; *see also* GOOG-PLAY-001184813 at GOOG-PLAY-001184823 (2015 internal Google presentation noting Play Store revenue share agreements give carriers "~15% of DCB related consumer App spend (prev. 27% DCB, 25% CC)").

202. GOOG-PLAY-004235359 at GOOG-PLAY-004235360 (emphasis in original).

203. *Id.*

App Distribution Market despite the fact that Google has been recouping its initial losses for almost a decade.

94. In short, once indirect network effects kicked in, granting Google an insurmountable monopoly in the Android App Distribution Market, Google was able to discontinue those incentives without fear of losing market share. As Google executive Jamie Rosenberg later commented: "We cut carriers in to *disincentivize* building their own stores and fragmenting the ecosystem. It worked."[204] Economists recognize how much an early, "first-mover" advantage means to incumbents in the context of network effects.[205]

2. Google's Exclusionary Restraints on OEMs

95. I understand that three contracts typically govern the relationship between Google and OEMs. First, OEMs enter Android Compatibility Commitments ("ACCs"), which replaced Anti-Fragmentation Agreements ("AFAs").[206] Broadly speaking, it is my understanding that AFAs and ACCs prevent OEMs from selling any "forked" Android-based devices that do not meet Google's compatibility standards. Provisions in a typical AFA read as follows:

- Company will not take any actions that may cause or result in the fragmentation of Android.
- Company will only distribute Products that are either: (i) in the case of hardware, Android Compatible Devices; or (ii) in the case of software distributed solely on Android Compatible Devices.
- Company may not distribute a software development kit (SDK) derived from Android, or derived from Android Compatible Devices.[207]

96. Record evidence indicates Google considers forks "a huge strategic headache for Google," because they could allow third parties to "grant exclusivity on an Android device to a competitor" or to promote a competitor by preloading its services.[208] OEMs must agree to an AFA (or now an ACC) in order to enter into MADAs, discussed in detail in the following subsection. Both are required to gain access to GMS, which has become critical for many apps to function.[209]

---

204. GOOG-PLAY-000439987.R at GOOG-PLAY-000440012.R (emphasis added).

205. *See, e.g.*, Agam Gupta et al., *Combating incumbency advantage of network effects: The role of entrant's decisions and consumer preferences*, 20(1) COMPETITION AND REGULATION IN NETWORK INDUSTRIES 3-32 (2019); Marvin B. Lieberman & David B. Montgomery, *Conundra and progress: Research on entry order and performance*, 46 LONG RANGE PLANNING 312–324 (2013), *available at* www.sciencedirect.com/science/article/abs/pii/S0024630113000344?via%3Dihub.

206. GOOG-PLAY-000127155 (Standard AFA Agreement signed by [redacted]); GOOG-PLAY-000808433 ([redacted]); GOOG-PLAY-000808062 ([redacted]); GOOG-PLAY-000808451 ([redacted]); GOOG-PLAY-003604523 ([redacted]); GOOG-PLAY-000416448 ([redacted]).

207. GOOG-PLAY-000127155 (Standard AFA Agreement signed by [redacted]).

208. GOOG-PLAY-001559464 at GOOG-PLAY-001559469.

209. I understand that Professor Schmidt's findings support this conclusion. *See also* GOOG-PLAY-001559473 ("Can partners sign just the ACC or just the MADA? I guess but this would be kind of pointless, since they need both to get GMS.").

Finally, most OEMs enter into Revenue Sharing Agreement ("RSAs")—whereby Google shares revenue it earns on the device with the OEM.[210]

a. Google's Mobile Application Distribution Agreements Require Distribution and Prominent Placement of The Play Store

97. Google owns some of the most highly valued and widely used Android Apps, including Google Search, Play Store, Maps, Chrome, Gmail, and YouTube. Yet Google refuses to make these apps individually available to OEMs for pre-installation, instead requiring OEMs to pre-install an entire GMS suite or forgo installation of *any* Google proprietary app.[211] It is all or nothing.[212] As explained in the Majority Report of the Congressional Subcommittee on Antitrust, Commercial and Administrative Law:

> Only through Google's licensing agreements can smartphone manufacturers access Google's proprietary apps, such as Gmail, YouTube, Chrome, Google Maps, and Google Play Store. In return, Google requires that certain apps must be pre-installed and must receive prominent placement on mobile devices.[213]

Google recognizes the importance of GMS: "Smartphones aren't very useful without an app store, map app, etc. OEMs need to either license those services through Google or else bear the expense of developing their own services."[214]

98. In addition, Google places a number of proprietary APIs in Google Mobile Services. I understand that Professor Schmidt finds that, without access to those APIs, a mobile device will lack access to many commercially important applications. In order to access these critical APIs so that applications can work, and to access certain highly demanded applications, OEMs must sign a MADA with Google to obtain the entire suite of GMS Apps.[215]

---

210. *See, e.g.,* GOOG-PLAY-002604372 ([redacted].

211. Google's MADA refers to these apps as "Core Applications." *See, e.g.,* GOOG-PLAY-000808375 at GOOG-PLAY-000808377 (2018 Motorola MADA); GOOG-PLAY-004552342 at GOOG-PLAY-004552344 (2017 Samsung MADA § 1.16 including Search, Chrome, Gmail, Maps, YouTube, and the Play Store). In addition to the "Core Applications," some MADAs require installation of region-specific apps called "Flexible Applications." GOOG-PLAY-004552342 at GOOG-PLAY-004552344 and GOOG-PLAY-004552347 (§§ 1.15, 3.3). Earlier MADAs defined the applications simply as "Google Applications." *See, e.g.*, GOOG-PLAY-000620996 § 1.1 (2011 Archos S.A. MADA). *See also* Android GMS, *The best of Google, right on your devices*, *available at* www.android.com/gms/.

212. GOOG-PLAY-003776161.R at GOOG-PLAY-003776177.R (discussing MADA as of 2015: "All of the 11 core apps must be pre-loaded...OEM's cannot pick apps a la carte.").

213. Investigation of Competition in Digital Markets: Majority Staff Report and Recommendations, H. SUBCOMM. ON ANTITRUST, COMMERCIAL AND ADMINISTRATIVE LAW OF THE COMM. ON THE JUDICIARY, (Oct. 6, 2020), at 212 [hereafter *Majority Staff Report*]. *See also* GOOG-PLAY-000400751 at GOOG-PLAY-000400773.

214. GOOG-PLAY-001559464.R at GOOG-PLAY-001559471.R. Google recognizes that "the smartphone business is very competitive: two players, Apple and Samsung, make the lion's share of profits while others operate on very thin or negative margins." *Id*. Licensing GMS, along with revenue shares, can be a "meaningful part of their business," and effectively leave OEMs with no choice but to take Google's conditions. *Id.*

215. GOOG-PLAY-001559464 at GOOG-PLAY-001559473; *see also* Rosenberg Dep. at 189:24-191:1 (testifying that developers write to "APIs that Google provides" and that "if it is an API that is distributed on with, you know, with that package of software on an Android device, then I don't know for certain, but I don't believe that that API would be available on a device that doesn't have Google Mobile Services by definition").

99. A typical MADA includes several clauses that require an OEM to give preference to Google Search and other applications in the GMS suite, including the Play Store. GMS Apps must be pre-loaded on devices and prominently displayed on the home screens; Google Search must be the default search engine and must also be given prominent placement.[216] The current MADA also requires OEMs to locate the Play Store on the home screens of each mobile device.[217] If an OEM wishes to install just one App from the GMS suite, the MADA requires that the "full suite of apps, services, and APIs" also be installed, and the number of required Google Apps has increased over time.[218] Collectively, all of the Apps in the GMS suite occupy valuable space on each user's mobile device that otherwise could be occupied by competing app stores or other Apps. In certain agreements, [redacted][219] In sum, Google is able to leverage the dominant positions of the Apps in the GMS suite to impose restrictions that are designed to further establish and protect its market power in the Android App Distribution Market.[220]

100. Although Google's MADAs do not prevent OEMs from preloading alternatives to GMS,[221] Google exploits the Play Store's prominent status, which works to the detriment of rival app stores, including any pre-installed near Google's app store.[222] Google's documents recognize

216. *See, e.g.,* GOOG-PLAY-000025345 at GOOG-PLAY-000025353-GOOG-PLAY-000025354. *See also* GOOG-PLAY-000400751.R at GOOG-PLAY-000400773.R ("Recent versions of the MADA require the [redacted] to appear on the default home screen.").

217. *See, e.g.*, GOOG-PLAY-004552342 at GOOG-PLAY-004552347 (§ 3.3: "preload on the Default Home Screen … the Google Play Client") ([redacted] MADA); GOOG-PLAY-000808375 at GOOG-PLAY-000808384 (§ 4.4: "[redacted]). Google's MADA has become more restrictive on where Play must be placed over time. Prior to 2014, the MADA did not require Android Marketplace, the predecessor to Play, to appear on the home screen. GOOG-PLAY-000620996 at GOOG-PLAY-000621002 (§ 10.2(a): "Google Search Widget and Android Market Client are presented to the End User on the panel immediately adjacent to the Default Home Screen.") (2011 MADA between Archos SA and Google Ireland Limited).

218. *See* GOOG-PLAY2-000001992 at GOOG-PLAY2-000001995 ("If a MADA partner chooses to install a GMS app on a device, the MADA requires the partner to install the full suite of Google GMS apps, services, and APIs on the device in question.").

219. *See, e.g.* GOOG-PLAY-004552342 at GOOG-PLAY-004552348 (§ 3.4) [redacted]; GOOG-PLAY-001404176 (noting that "most users just use what comes on the device. People rarely change defaults. This underpins our Toolbar and browser chrome distribution models, including iPhone, with data to prove its success").

220. Google's documents establish that the MADA requirements are essential to the Play Store's dominance. GOOG-PLAY-006355073 ("Fortunately, we'll always have the placement/pre-install advantage which is 90% of the battle."); GOOG-PLAY-004494430.C at GOOG-PLAY-004494433.C ("Google app distribution to get max usage for minimum cost" is one of four issues "MADA seeks to balance").

221. *See* GOOG-PLAY2-000001992 ("Google's MADAs also do not prevent signatories from preloading an alternative to GMS.").

222. Samsung's Galaxy Store is an example of this. As discussed below in Part IV.C., while the Galaxy Store came pre-installed along with the Play Store on the Galaxy S10 and later models, Google has engaged in a course of conduct designed to discourage effective competition to the Play Store from Samsung. Google has recognized its efforts were effective, pronouncing that "cannibalization of Play store revenue due to Galaxy store" as "none to minimal." *See* GOOG-PLAY-000443908 at GOOG-PLAY-000443909. More broadly, multiple studies have shown that defaults can impact consumer decisions towards choices such as retirement savings, energy use, and eating healthier foods. *See* Brigitte C. Madrian & Dennis F. Shea, *The Power of Suggestion: Inertia in 401(k) Participation*

the value of the Play Store's preferential placement on the home screens of mobile devices, including its importance to Samsung: "[redacted]"[223] And in questioning whether users and developers would really choose the Play Store, given a choice, a high-level Google employee wrote, "[redacted]"[224]

b. Google Has Deployed Multiple Measures to Ensure That Amazon Would Not Become an Effective Play Store Competitor

101. Google has taken multiple steps to stop competitors from succeeding with a competing app store. Amazon in particular was a potential competitor that has been substantially foreclosed by Google's conduct, which raised the costs to Amazon of competing with its rival app store. *First*, as discussed above, the MADAs mandate installation of the Play Store as a condition of installing any App in the GMS suite. In 2014, Amazon launched a bare Android device called the "Fire Phone," which was not pre-loaded with any of the GMS Apps. Indeed, users were "locked out" by Google from downloading these Apps. Unsurprisingly, consumer demand for a device that cannot include Apps like YouTube, Gmail, or Google Maps was low, and Amazon discontinued the device within a year.[225] The MADAs prevented an OEM from customizing the Apps on mobile devices by precluding an alternative bundle comprised of a rival app store (including Amazon's app store) alongside Google's other popular (non-Play Store) Apps—that is, a rival app store would need to compete across every dimension of Google's app suite at once, effectively raising its costs. Without a successful "Fire Phone" due to Google's restrictions, Amazon was less likely to fully compete in the Android App Distribution Market by investing and developing a mobile app store that would rival the Play Store in scope and reach.[226]

102. *Second,* in 2015 Amazon released a backdoor to the Amazon Appstore (Amazon's App distribution store) through the Amazon App (Amazon's shopping App) that was available for download on the Play Store. As one media outlet noted, "The move effectively turns Amazon's flagship application—an app that has somewhere between 50 million and 100 million installs, according to Google Play's data for the smartphone version—into an app store app that directly

---

and Savings Behavior, 116(4) QUARTERLY JOURNAL OF ECONOMICS 1149-1187 (2001); Zachary Brown, Nick Johnstone, Ivan Haščič, Laura Vong, and Francis Barascud, *Testing the effect of defaults on the thermostat settings of OECD employees*, 39 ENERGY ECONOMICS 128-134 (2013); John Peters, Jimikaye Beck, Jan Lande, Zhaoxing Pan, Michelle Cardel, Keith Ayoob, and James O. Hill, *Using Healthy Defaults in Walt Disney World Restaurants to Improve Nutritional Choices*, 1(1) JOURNAL OF THE ASSOCIATION FOR CONSUMER RESEARCH 92-103 (2016).

223. GOOG-PLAY-000832471.

224. GOOG-PLAY-000292207.R at GOOG-PLAY-000292226.R (*see also* GOOG-PLAY-000292207.R at GOOG-PLAY-000292213.R; GOOG-PLAY-006355073).

225. Edelman & Geradin at 167 (citing www.wsj.com/articles/amazon-fire-phone-review-full-of-gimmicks-lacking-basics-1406077565).

226. *See* GOOG-PLAY-001317740 at GOOG-PLAY-001317741 (2011 Google summary of competing app stores: "*Amazon making play at mobile content distribution* with launch of Appstore for Android… *branded headseat (Amazon Blaze) on the horizon*.") (emphasis in original); GOOG-PLAY-001451619 (Google negotiators recommend rejecting any requests from Amazon to modify MADA for Fire devices, knowing GMS would make the device stronger but competing app stores would cause "fragmentation."); *see also* GOOG-PLAY-007657997 at GOOG-PLAY-007658010 (2017 concern that although Amazon's app store "lacks critical mass of users and developers today," "[i]f they can achieve either, we believe this will create a virtuous cycle drawing in more users and developers – increasing appeal of Fire devices and greatly increas[ing] the severity of the threat.").

competes with Google Play, while also being sold on Google Play."[227] Google quickly forced Amazon to update its app to remove this functionality.[228]

103. *Third*, using its control over access to the GMS suite and the dominant position of the Play Store in the Android App Distribution Market, Google was able to introduce additional restraints that strongly discouraged use of bare Android devices. For example, consumers who had purchased an App via the Play Store were prohibited from re-downloading that App to a bare Android device that could not include the Play Store. These consumers would have to repurchase the same App on their bare device to keep using it.[229] Accordingly, consumers wishing to move to non-Google Android devices, such as the Amazon "Fire" phone, would be required to repurchase all Apps they had previously purchased from the Play Store or contact the developer directly to request a free download on the new device.

c. Google Discouraged Samsung from Effectively Competing with the Play Store in the Distribution of Apps in the Android App Distribution Market and Entered into Deals with Developers to Mitigate the Risk of Competition from Samsung

104. Google's treatment of Samsung, the largest Google Android OEM, illustrates Google's recognition of the potential competitive threat posed by a competing app store and the lengths Google would take to avoid such competition. Google engaged in numerous detailed, strategic programs to "mitigate challenges" posed by Samsung's expansion of its app store,[230] "Samsung Apps," later rebranded as the "Galaxy Store."

105. As early as 2011, Google recognized that competition from a Samsung app store could be "destructive" to Google Android because it would lead to "developer exclusives, competing business models, etc."[231] Google also feared that Samsung could secure exclusivity of popular new Apps meaning that "the app is available on Samsung stores or devices and not on Play for a period of time, usually months."[232] Pre-loading of the app store on all Samsung devices would allow Samsung to "shift consumer behavior away from shopping for apps and games in the Play Store. This fundamentally impacts Play store effectiveness."[233] Google's goal was: "[G]et

---

227. GOOG-PLAY-000832219 at GOOG-PLAY-000832221.

228. Google's Jamie Rosenberg made this clear in an email to Sameer Samat on March 14, 2015. GOOG-PLAY-000832219 ("New downloads of the Amazon Mobile app (as of 12/12) would not have App Store functionality.").

229. Edelman & Geradin at 167 (citing www.wsj.com/articles/amazon-fire-phone-review-full-of-gimmicks-lacking-basics-1406077565). The Nokia X phone, also launched in 2014, met a fate similar to the Amazon Fire Phone for these reasons. *Id*.

230. GOOG-PLAY-000004762.R at GOOG-PLAY-000004764.R; *see also* GOOG-PLAY-000367346.R.

231. GOOG-PLAY-006359924 at GOOG-PLAY-006359925 (explaining "we don't believe Samsung should be cultivating its own developer ecosystem").

232. GOOG-PLAY-006359924.

233. GOOG-PLAY-006359924 at GOOG-PLAY-006359925.

them to stop distributing apps through Samsung App store"[234] or otherwise "duplicat[ing] our services on Android."[235]

106. [redacted] [239]

107. [redacted]

234. GOOG-PLAY-001438299 at GOOG-PLAY-001438300. *See also* GOOG-PLAY-004253884 at GOOG-PLAY-004253907 ("Samsung is competing directly with Google, positioning itself to control more consumer touch points and to enhance control over the home-screen").

235. GOOG-PLAY-001449339 at GOOG-PLAY-001449340 (explaining "Samsung's duplication of our services on Android" was "one of the critical issues with the partnership right now" and the presence of the "Samsung Apps relative to Google Play" was the "most glaring" example of such duplication).

236. GOOG-PLAY-000417080.

237. GOOG-PLAY-001455849 (explaining the Samsung App store had "grown to a store with 70k apps," but it had not done well).

238. *Id.*

239. GOOG-PLAY-001847447 at GOOG-PLAY-001847448 (explaining "Galaxy Apps is a parking place for 'less than 1,000 apps' that showcase Samsung devices in some way.").

240. GOOG-PLAY-004509271 at GOOG-PLAY-004509272.

241. *Id.*

242. *Id.*

243. GOOG-PLAY-000004762 at GOOG-PLAY-000004785. *See also* GOOG-PLAY-000367346.

244. The technical processes known as "Alley-oop" was a critical part of Google's offering. As described by Google, "Alley-oop" meant that Google provided the "delivery infrastructure for Samsung's Galaxy Store" whereby any download would appear to the user as occurring "through Play without leaving the Galaxy Store." GOOG-PLAY-000464148 at GOOG-PLAY-000464150. It would also "apply to Game Launcher and any other Samsung product where apps can be downloaded through the Galaxy Store today." *Id.* In practical terms, that meant Google would continue to receive its 30 percent share of any paid App download through the Galaxy Store, and Samsung could not entice developers with lower take rates. *Id.* at GOOG-PLAY-000464149-GOOG-PLAY-000464150.

[redacted][245]

108. Google's Jamie Rosenberg confirmed that the effect of Google's Project Banyan proposal to Samsung would have been that Samsung and Google would not compete for the time of developers' engineers, and would not have competing commercial terms for developers.[246]

109. But Project Banyan alone was not enough to address Google's ongoing fears of competition from Samsung. Google remained concerned that:

- Samsung could drive more developers and users to its store by discounting its rev share to developers as "payment" for exclusives and unique content, and by using "[p]romotions and discounts to users to use their store using notifications, emails, etc."[247]

- Samsung could respond "with a very public and disruptive rev share model (i.e., it just decides that it will take only 5% and use its app store for purposes of building [form of payments] and user profiles and differentiating devices.[)]"[248]

- "[I]f Samsung wins the hearts & minds of developers on this, it could create enormous pressure on us to unblock their opportunity one way or another," and while developers "will tolerate some premium pricing for distribution through Play / Google and all that we provide, but not a gap that wide."[249]

d. Google Project Hug Secured Content from Some of the Largest Developers, Preventing them from Giving Competing Stores the Exclusive Content Necessary to Help Drive Usage

110. Google also introduced "Project Bear Hug," later shortened to "Project Hug," which imposed a contractual requirement on major Play Store developers to prevent app stores from entering "exclusives for the most lucrative and risky developers."[250] Known internally as the [redacted] requirement, Project Hug required developers [redacted], thereby foreclosing the ability of developers to enter into lucrative [redacted] with other app stores.[251] Google's documents indicate that Projects Banyan and Hug worked together to enable Google to continue to reap the rewards of its supra-competitive take rates, by (1) limiting Samsung's Galaxy Store catalog, which minimized the effectiveness of any "deep discounts" Samsung might offer;[252] and (2) drafting contract terms that "[d]isincentivize broad discounts (across many / all apps to siphon engagement away from equivalent Play app)."[253]

---

245. GOOG-PLAY-000367346.R at GOOG-PLAY-000367347.R.

246. Rosenberg Dep. at 114:24-115:22; *see also* PX 783, GOOG-PLAY-007384816 at GOOG-PLAY-007384818.

247. GOOG-PLAY-001877016 at GOOG-PLAY-001877020–GOOG-PLAY-001877021.

248. *Id*. at GOOG-PLAY-001877020.

249. *Id*.

250. GOOG-PLAY-001877016 at GOOG-PLAY-001877019.

251. GOOG-PLAY-000000807 at GOOG-PLAY-000000810.

252. GOOG-PLAY-000464148 at GOOG-PLAY-000464151.

253. *Id*.

111. One of the key Google employees responsible for implementing Project Hug has admitted that the terms of the Project Hug agreements, which prevented developers from [redacted], including Samsung's Galaxy Store, have effectively mitigated the competition from Samsung. Mr. Koh explained that Project Hug's [redacted] was aimed at ensuring "developers were prioritizing Google Play when they were thinking about launching a new game or a major new content update"[254] and that Google was concerned that developers might strike deals to [redacted] through Samsung's Galaxy Store.[255] Project Hug prevented those developers, however, from [redacted] on Samsung Galaxy or other competing stores. Mr. Koh viewed Project Hug as "mitigating our risk of losing out to competition" from Samsung and other competitors.[256] In this way, Project Hug impaired competition from other stores.

112. [redacted][257]

113. While Google "halt[ed] work" on Project Banyan in July 2019 and instructed its employees to "not proceed with the related work streams involving our respective app store,"[258] its efforts to minimize competition from Samsung's Galaxy Store had already been effective. In 2019, Google executives presented a "Galaxy Store Performance Update" concluding that the Galaxy Store was unable to meaningfully compete with the Play Store as the Galaxy Store "continues to lack compelling differentiation in terms of new exclusive titles or [In-App Products],"[259] and that "the cannibalization of Play store revenue due to Galaxy store is none to minimal."[260] This result stemmed from the "firewall that Galaxy store is limited to promotion and curation only for Samsung exclusives" and Samsung's reluctance to "start conflicting with Play curation."[261]

114. Google's subsequent analysis of a major game release, [redacted] confirmed the Update's findings and estimated the Galaxy Store grossed only [redacted] to [redacted] (across all apps), or between [redacted] and [redacted] percent of the amount consumers spent on the same Apps on the Play Store.[262] Thus, Google's pressure and its Project Hug agreements thwarted

---

254. Koh Dep. at 362:22-363:1.
255. *Id.* at 364:13-365:4.
256. *Id.* at 367:4-16.
257. GOOG-PLAY-002994573 at GOOG-PLAY-002994574.
258. GOOG-PLAY-004136427.
259. GOOG-PLAY-000443908.R at GOOG-PLAY-000443911.R.
260. [redacted]
261. GOOG-PLAY-000367346.R at GOOG-PLAY-000367351.R.
262. GOOG-PLAY-000001317. In a 2019 internal email referencing "top games" such as [redacted] Google's Christian Cramer asked, "[a]re we tracking how much traction these top games get on other platforms such as the Galaxy store?" In response, Google's Brian Brazinski reported that

Samsung's efforts to differentiate its product offerings through exclusives with developers offering unique content at lower prices. Otherwise, more developers and consumers would have been attracted to the Galaxy Store. Google was able to achieve this result even though Samsung was a sophisticated, well-resourced market entrant that secured broad pre-installation of its own app store.

e. Google's New Revenue Sharing Agreements With OEMs Are Designed To Further Entrench The Play Store's Monopoly

115. Google has further insulated the Play Store from competition through its most recent series of OEM agreements. Stylized as [redacted] on the absence of any pre-installed third-party apps that compete with Google apps, including third-party app stores.[263]

116. I understand that Google chose to enter into these agreements after certain OEMs had begun to build out distribution scale that could allow third-party stores to compete with the Play Store if successfully preloaded.[264] These Premier Tier payments thus compensated OEMs on [redacted] if the OEM agreed not to allow third-party app stores to be preinstalled on each Premier Tier device.[265] The RSA payments offered by Google were economically significant, as Google expected total spend to be nearly [redacted] in 2020 and up to [redacted] in 2023.[266] Indeed, Google's stated goal was to move all [redacted].[267]

117. Google's RSA agreements prohibiting the pre-installation of competing app stores affected a growing and substantial portion of Android devices shipped. Google has entered into RSA agreements with at least [redacted] OEMs to date.[268] In 2020, based on projections for [redacted] OEMs who had then signed Google Forward agreements, Google anticipated [redacted] Premier Tier devices to ship in [redacted] and [redacted].[269] According to Google's internal estimates, as of January 2021, [redacted] out of [redacted] devices sold were "Premier Devices," and this was expected to increase in

---

Monthly Active User ("MAU") and Daily Active User ("DAU") metrics for the Galaxy Store were only a small fraction of those for the Play Store ("MAU/DAU for both HPWU and PG are [redacted] of Play DAU/MAU"). *Id.* at 318. Similarly, "Total hours played on Galaxy apks [redacted] of game time on Play apk." *Id.* More broadly, the same document also estimated that aggregate consumer spend for Apps and Games ("A&G") in the Galaxy Store was about [redacted] to [redacted] percent of those in the Play Store. *Id.* at 317 ("Galaxy Store grosses [redacted] today in A&G consumer spend (between [redacted] of Play A&G spend[.]").

263. *See* GOOG-PLAY-004489416.R; GOOG-PLAY-000443763; GOOG-PLAY-004494430.C; GOOG-PLAY-004486928.R.

264. GOOG-PLAY-000443763 at GOOG-PLAY-000443773.

265. *Id.* at GOOG-PLAY-000443775.

266. GOOG-PLAY4-007239946.

267. *Id.*

268. GOOG-PLAY-000620210 ([redacted]); GOOG-PLAY-000620638 ([redacted]; GOOG-PLAY-005706338 ([redacted]); GOOG-PLAY-008111867 ([redacted]); GOOG-PLAY-001745614 ([redacted]); GOOG-PLAY-000416708 ([redacted]); GOOG-PLAY-000620282 ([redacted]); GOOG-PLAY-000620442 ([redacted]); GOOG-PLAY-000620131 ([redacted]); GOOG-PLAY-005706436 ([redacted]); GOOG-PLAY-005706676 ([redacted]); GOOG-PLAY-007038477 ([redacted]); GOOG-PLAY-007038511 ([redacted]); GOOG-PLAY-000620478 ([redacted]); GOOG-PLAY-005706728 ([redacted]); GOOG-PLAY-000416651 ([redacted]).

269. GOOG-PLAY-008006134.

[redacted][270] For some OEMs, all or nearly all of their devices sold in January 2021 were "premier tier," demonstrating the power of Google's revenue sharing terms.[271]

118. The RSA agreements substantially foreclose some of the remaining and most viable distribution avenues for competitive stores. For these agreements, Google targeted those groups of OEMs that were building, according to its internal documents, "significant distribution scale." Google planned to "incentivize OEMs to prioritize Play" with these agreements.[272] "Prioritizing Play" meant preloading "Play as the exclusive app store on devices."[273] As of 2019, Google's plan was to move "all current non-Samsung RSA partners" to its RSA agreements.[274] Google's internal documents indicate that it focused on "non-Samsung RSA partners" because Samsung was itself investing in its own Galaxy Store.[275] Assuming Google succeeded in moving most non-Samsung manufacturers to this program over the last couple of years, it would have cut off virtually every important pre-loading distribution avenues for competitive third-party app stores, substantially foreclosing their ability to compete in the relevant market. These new, broad restrictions on potential third-party competitors serve to further entrench the Play Store's dominance.

3. Google's Exclusionary Android App Distribution Market Restraints on App Developers

119. Google's agreements with developers inhibit competition from rivals in the Android App Distribution Market by prohibiting the distribution of competing app stores through the Play Store and by prohibiting developers from steering users to lower-priced App distribution channels or using user information learned through the Play Store. Developers are precluded from using the Play Store "to distribute or make available any Product that has a purpose that facilitates the distribution of software applications and games for use on Android devices outside of Google Play."[276] Nor can developers steer consumers to other platforms or websites to purchase or download Apps or In-App Content: "You may not use user information obtained via Google Play to sell or distribute Products outside of Google Play."[277]

---

270. GOOG-PLAY-003894142 at GOOG-PLAY-003894172. I understand that Plaintiffs do not have updated discovery from Google to determine the number of current devices that ship under Premier Tier terms.

271. *Id.* at GOOG-PLAY-003894173 (noting [redacted] percent premier tier for Sony, [redacted] percent for [redacted], and [redacted] percent for [redacted]).

272. *See* GOOG-PLAY-000443763 at GOOG-PLAY-000443774.

273. *Id.* at GOOG-PLAY-000443775.

274. GOOG-PLAY4-007239946 at GOOG-PLAY4-007239947.

275. GOOG-PLAY-004502766.

276. *See* Google Play Developer Distribution Agreement (as of Jan. 4, 2014) ("You may not use the Market to distribute or make available any Product whose primary purpose is to facilitate the distribution of software applications and games for use on Android devices outside of the Market."); Google Play Developer Distribution Agreement (as of Sep. 25, 2014) ("You may not use the Store to distribute or make available any Product which has a purpose that facilitates the distribution of software applications and games for use on Android devices outside of the Store."); Google Play Developer Distribution Agreement (effective as of June 12, 2020) ("You may not use Google Play to distribute or make available any Product that has a purpose that facilitates the distribution of software applications and games for use on Android devices outside of Google Play."). For years, dating at least back to 2008, Google called this the "Non-Compete" provision. *See, e.g.*, GOOG-PLAY-000054021 at GOOG-PLAY-000054022; GOOG-PLAY-000054683 at GOOG-PLAY-000054685 (2008 version). By 2014, Google had dropped this label in favor of calling the provision "Alternative Stores." Compare GOOG-PLAY-000054039 at GOOG-PLAY-000054041 (2014 version) with GOOG-PLAY-000053975 at GOOG-PLAY-000053977 (2017 version).

277. GOOG-PLAY-000053875 at GOOG-PLAY-000053876 (Google Play Developer Distribution Agreement (effective as of Nov. 17, 2020), point 4.9).

120. In addition, to access Google's App Campaigns program, Android App developers must list their Apps in the Play Store.[278] Only Apps that were distributed in the Play Store could participate in Google's App Campaign program, a program specifically designed to allow developers to place ads for Apps and In-App Content on Google's self-proclaimed most valuable properties.[279] Those "properties," which are specially optimized for the advertising of mobile Apps, included Google Search, YouTube, Discover on Google Search, and the Google Display Network. Google was explicit about this linkage in its marketing, representing that placement in the Play Store enabled developers to "get your app into the hands of more paying users" by "streamlin[ing] the process for you, making it easy to promote your apps across Google's largest properties."[280] This conduct further entrenched Google's monopoly in the Android App Distribution Market by coercing developers to list their Apps in the Play Store or risk losing advertising access to some of the Internet's most effective advertising space.

121. A key part of Google's effort to prevent developers from supporting a rival app store was Project Hug.[281] Like Google's payments to OEMs and mobile operators, Project Hug followed a simple formula: pay the potential competitor enough to prevent it from going "off-Play" or from giving a competitor such as Samsung exclusive content, but less than Google's expected loss from damage to its monopoly.[282]

122. Project Hug provided incentives in the form of [redacted][283] The monetary payments were significant, typically

---

278. Google Ads Help, *About App campaigns, available at* support.google.com/google-ads/answer/6247380?hl=en.

279. *See* GOOG-PLAY-000226999 at GOOG-PLAY-000226999-GOOG-PLAY-000227001 (Co-marketing fund agreement) [redacted]

280. Google Ads Help, *About App campaigns, available at* support.google.com/google-ads/answer/6247380?hl=en.

281. GOOG-PLAY-000445443.R at GOOG-PLAY-000445458.R. Externally, in negotiations with developers, Google referred to Project Hug as the "Games Velocity Program" ("GVP"). *See* GOOG-PLAY-000932349. Google required developers who signed GVP agreements to keep the deal confidential. *Id.* at GOOG-PLAY-000932352 ("Neither party may make any public statement regarding this Addendum without the other's written approval.").

282. [redacted] *Id.* at GOOG-PLAY-000000809. In return, [redacted]

283. *See, e.g.,* GOOG-PLAY-000559379 at GOOG-PLAY-000559382 and GOOG-PLAY-000559384; GOOG-PLAY-000000807; GOOG-PLAY-000229696; GOOG-PLAY-005027813 (showing Hug-approved budget,

amounting to around [redacted] to [redacted] percent of total consumer spend on the Play Store.[284] In return for these payments, Google required that developers invest in the Play Store [redacted][285]

123. Google's Project Hug and the [redacted] requirement in particular proved successful. Within one year, [redacted] of the top [redacted] game developers had signed Games Velocity Program Agreements.[286] One of the holdouts, [redacted]t, a popular gaming developer, said that it was considering launching its own "Off-Play Webstore" to avoid Google's commission charge.[287] According to Google, if [redacted] were to migrate all of its spend off Google, the impact would be "[redacted] to [redacted] in spend from 2019-22."[288] Google, aiming to "[m]itigate the risk" of [redacted] defection from Google Play, made an offer to [redacted] that involved revenue-sharing incentives and "reinvesting" customer spend into mutually beneficial marketing activities.[289] By April 2020, Google had offered [redacted] a "[redacted]"[290] Shortly thereafter, [redacted] signed a Project Hug agreement.[291]

124. Altogether, Google estimated that Project Hug resulted in "[redacted] Play margin loss risk mitigated (2019-2022)," while also driving an additional [redacted] in cross-platform revenue.[292] Google also noted that signed Project Hug developers had stopped "escalat[ing] about rev share to date" and were "leaning into Play & Google partnership."[293] In 2020, Google decided to double down and rolled out the Games Velocity Program 2.0, offering Project Hug-like benefits to the next [redacted] largest game developers.[294]

---

developers and their expected 2020 Play spend, along with [redacted]

284. GOOG-PLAY-000559379 at GOOG-PLAY-000559382-GOOG-PLAY-000559383 ("In a typical deal, Play reinvests [redacted]. *See also* GOOG-PLAY-004119228.R at GOOG-PLAY-004119237.R ("Effectively, GVP is a cross-Google deal structure where by [*sic*] Play takes [redacted] of the revenue share a developer would have owed Google and allows the developer to reinvest those dollars towards Google cloud credits, incremental [redacted]

285. GOOG-PLAY-000000807 at GOOG-PLAY-000000810-GOOG-PLAY-000000811.

286. GOOG-PLAY-000001976.

287. *Id.* ("Tencent is therefore considering [whether] to launch their subscription service in an off-Play Webstore to get the economics they aspire.").

288. GOOG-PLAY-000003283.R at GOOG-PLAY-000003308.R.

289. GOOG-PLAY-000001976 at GOOG-PLAY-000001976-GOOG-PLAY-000001977 ("As [redacted] has influence over a large portfolio of games developers that adds up to [redacted] of Play revenue, [redacted] would have a realistic opportunity to successfully launch an off-Play business which would result in a margin risk of [redacted] for Play." The GVP (Games Velocity Program) deal was designed to "[m]itigate the risk" of [redacted] defection. It included an offer to share revenue (percentage of [redacted] and [redacted]) and to "[redacted]" a percentage of subscription spend into "[redacted]").

290. GOOG-PLAY-000003283.R at GOOG-PLAY-000003286.R.

291. GOOG-PLAY-003899355.R at GOOG-PLAY-003899360.R.

292. GOOG-PLAY-004146689.R at GOOG-PLAY-004146697.R.

293. *Id.*

294. GOOG-PLAY-004146689 at GOOG-PLAY-004146722.

125. Google's "sim-ship" requirement that game developers release all games on the Play Store on the game's day of launch can be understood as a most-favored-nations ("MFN") clause foreclosing competing app stores from entering into exclusive arrangements with developers whereby, in return for a substantial payment, the developers agree to launch a title on a rival app store. When employed by dominant platforms such as Google, MFNs have been recognized as impairing competition.[295] Pricing-parity MFNs preclude the supplier (developer) from pricing below the price it charges on the platform,[296] but MFNs can also dictate non-price terms such as product quality or timing that indirectly weaken price competition. Baker and Scott Morton explain that "[p]latform MFNs with greater scope and duration would be expected to have stronger anticompetitive effects and impose larger penalties[.]"[297] The scope of Google's equivalent to an MFN here can be measured by the revenue-share of the participating Apps among U.S. customers. Using Google's transaction data, I estimate that the [redacted] developers that were part of Project Hug in 2019[298] accounted for [redacted] percent of all game-related App revenue and [redacted] percent of all App revenue on the Play Store in 2019. I also estimate that the [redacted] game developers that were part of Project Hug and the expanded Games Velocity Program 2.0 in 2020[299] accounted for [redacted] percent of all game-related App revenue and [redacted] percent of all App revenue on the Play Store in 2020. Put differently, Google's equivalent to an MFN prevented a potential or actual rival app store from preferentially or exclusively selling over [redacted] of the most popular applications in 2020.

126. Deposition testimony supports the conclusion that potential competition was eliminated. According to Google's Mr. Koh, the [redacted] requirement eliminated one potential avenue of competition from Samsung's Galaxy Store,[300] [redacted] For this reason, Mr. Koh "viewed it as it mitigating our risk of losing out to competition" from Samsung and other competitors.[301]

127. Google's sim-ship requirement effectively prevented any rival app store from offering consumers major gaming titles at an earlier date or on an exclusive basis—preferences that could have helped rival app stores to gain a foothold with consumers and convince OEMs to pre-install a rival app store. Once such a new app store was widely distributed, developers could have used that alternative distribution outlet to pressure Google to reduce its take rate in the Android App Distribution Market via the threat of steering. Thus, although the MFN equivalent

---

295. *See, e.g*, Jonathan B. Baker & Fiona Scott Morton, *Antitrust Enforcement Against Platform MFNs*, 127(7) YALE LAW JOURNAL 2176-2202, 2177 (2017) (studying the effects of MFNs under an "agency distribution model," whereby "the platform does not take ownership of the good (*e.g.*, the hotel room) but sells it on behalf of the owner at a price chosen by the owner.").

296. A pricing-parity requirement creates an incentive for the seller not to offer low prices because any price discount must be offered to all covered buyers, which makes discounting more expensive and thus softens price competition. *Id.* at 2179-2180.

297. *Id.* at 2182.

298. List taken from GOOG-PLAY-000237798. Tencent is excluded from this list. GOOG-PLAY-000001976 ("Tencent is therefore considering to launch their subscription service in an off-Play Webstore to get the economics they aspire.").

299. The Project Hug developer list was taken from GOOG-PLAY-000237798, with [redacted] excluded, *supra*. The expanded GVP developer list is taken from GOOG-PLAY-004146689 at GOOG-PLAY-004146710.

300. Koh Dep. at 364:9:365:4.

301. *Id.* at 367:4-16.

here was not aimed directly at prices, it ultimately reduced price-based competition by eliminating the prospect of steering.

4. Technical Barriers

128. I understand that Professor Schmidt finds that Google uses various technical barriers that prevent or inhibit competition from other app stores with its Play Store. Google makes it unnecessarily difficult for consumers to download Apps from rival app stores. I understand Professor Schmidt explains that, in many cases, users must first locate the store on the Internet, then sideload the store, and then change a security setting on Android devices, which Google discourages by first creating default settings that block these downloads, and then, if the user attempts to change the setting to download an application, by displaying often misleading warnings regarding competing app stores.

129. For example, in 2016 to download the Amazon app store, a user had to complete a series of 19 steps, including selecting "Unknown Sources" within the user's security settings, and navigating three separate security warnings.[302] The "Unknown Sources" label is ominous, with early versions warning users that downloading app stores would make your "[p]hone and personal data … more vulnerable to attack."[303] Android has used variations of this warning even for reputable stores like Amazon's.[304] I understand that Google has continued to use such warnings when a user attempts to install rival app stores.

130. I understand that Professor Schmidt finds that Google further frustrates the ability of consumers to customize their devices by imposing technical barriers that impact the downloading of Apps from outside the Google Play Store, including from developer websites, which must proceed through an "unknown sources" flow. While Google Android technically permits sideloading, I understand that Professor Schmidt will opine that Google has made it unnecessarily cumbersome. I also understand that the Google, over time, has increased the frequency with which a user encounters the "unknown sources" flow. In the past, a user would trigger the "unknown sources" flow when downloading a third-party app store, but not when thereafter downloading an app from that third-party app store. Now, a user triggers the "unknown sources" flow when downloading the third-party app store *and* when downloading the user's first app. As a result of these technical impediments, sideloading is not a commercially viable alternative distribution channel for most developers.[305]

---

302. A Google presentation recognized the significance of this sideloading deterrent by documenting the 19 steps required to successfully install Amazon Underground. GOOG-PLAY-000297309.R at GOOG-PLAY-000297311.R-GOOG-PLAY-000297314.R. I understand that this is confirmed by Professor Schmidt's findings.

303. Blake Stimac, *How to sideload an app onto your Android phone or tablet*, GREENBOT, (July 17, 2014), *available at* www.greenbot.com/article/2452614/how-to-sideload-an-app-onto-your-android-phone-or-tablet.html.

304. Jimmy Westenberg, *How to install the Amazon Appstore on your Android [Android 201]*, ANDROIDGUYS, (Apr. 5, 2014), *available at* www.androidguys.com/tips-tools/install-amazon-app-store-android/.

305. *See also* Jerry Hildenbrand, *Sideloading and Unknown Sources on Android: How to do it and fix errors*, ANDROID CENTRAL, (Apr. 16, 2020), *available at* www.androidcentral.com/unknown-sources ; Edelman & Geradin , *supra*, at 168 ("enabling sideloading requires first reducing phone security settings, which users will rightly hesitate to do."); Joel Snyder, *What are the risks of sideloaded Android applications?*, SAMSUNG KNOX, (Apr. 20, 2020), *available at* www.samsungknox.com/en/blog/what-are-the-risks-of-sideloaded-android-applications ; Dallas Thomas, *Get Easy Updates on Sideloaded Android Apps*, GADGET HACKS, (Dec. 27, 2016), *available at* android.gadgethacks.com/how-to/get-easy-updates-sideloaded-android-apps-0174291/.

131. In addition, I understand that Professor Schmidt will explain that Google has also historically restricted auto-updating capability to Apps listed in the Play Store or app stores pre-installed by OEMs. (Auto-updating is properly understood as a function in the Android App Distribution Market; there is no separate demand among consumers for that function apart from an app store.) This restriction inhibits competition by degrading the user experience for Apps downloaded from an alternative source. I understand that Professor Schmidt will set forth that Google only recently loosened this restriction when it released Google Android version 12 in October 2021.[306]

IV. GOOGLE'S ANTICOMPETITIVE CONDUCT IN THE IN-APP AFTERMARKET OF SERVICES IN SUPPORT OF THE PURCHASE OF IN-APP CONTENT

132. In this Part, I use data, economic methods, and evidence common to all Class members to establish that the global In-App Aftermarket for services in support of the purchase of In-App Content is a relevant market for antitrust purposes, that Google possesses significant market power in this market, and that Google has engaged in anticompetitive conduct to further and retain its power in this market.

A. The In-App Aftermarket Is a Distinct Relevant Antitrust Product Market

133. After a consumer has examined the available options on the Play Store, selected an App, and downloaded it to her device, the developer may choose to offer In-App Content for purchase and download. Such content can include, among others, access to an ad-free version of the App; videos or interactive programs that run within the App; or avatars, skins, or other accessories used for in-App gameplay. When Apps are free, the sale of In-App Content is often a major way for developers to earn revenue. Indeed, record evidence suggests that the revenues developers receive for In-App Content vastly exceed the revenues developers receive for initial download of Apps.[307]

134. I understand that to provide In-App Content, developers must be able to authorize the use of such content and collect payment from consumers for it. Payment systems require software that securely verifies and accepts customer purchases and may perform other related functions such as storing information about users and their purchasing history or tracking payment histories. Payment systems are also keyed to trigger the unlocking and authorization for the delivery of In-App Content once it is purchased and paid for by consumers. That is, distribution of In-App Content is not complete until the consumer can use those items, and that does not occur until payment is processed and the feature is unlocked.[308]

135. The Play Store is not needed in these In-App Aftermarket services, as the matchmaking function is not present. Thus, the one-sided In-App Aftermarket is distinct from the two-sided Android App Distribution Market. *See* Part III.A, *supra.* As Mr. Koh testified, in the delivery of In-App Content to consumers, in-app item inventory resides on developers' (not Google's) servers.[309] Although (by virtue of the billing system tie-in) Google provides

---

306. *See also* Google Play services, *available at* developer.android.com/distribute/play-services ; GOOG-PLAY-004904016.

307. GOOG-PLAY-000379093.

308. I understand that Professor Schmidt's findings support this conclusion.

309. Koh Dep. at 381:4-382:6.

confirmation of payment, it is the developer, and not Google, that releases and delivers In-App Content to the consumer.[310] I also understand that Professor Schmidt finds that, as a technological matter, there is no basis for Google to insert itself into the In-App Aftermarket by requiring that developers use Google Play Billing.

136. In Appendix 5, I use a critical-loss analysis to inform a SSNIP test, which shows that a hypothetical monopoly provider of In-App Aftermarket services could profitably raise prices above competitive levels demonstrating that the In-App Aftermarket is a separate relevant antitrust product market.

137. Google uses its contracts with developers to control the In-App Aftermarket. Initially, Google requires that, if an App is offered through the Play Store, the developer must use Google Play Billing for all subsequent sales of digital In-App Content that is consumed within the App.[311] Google utilizes Google Play Billing to impose a general take rate of 30 percent (with the aforementioned exceptions)—the same take rate it commands in the Android App Distribution Market—on all such purchases in perpetuity.[312]

138. Absent Google's restrictions, competition would materialize and there would be alternative providers of In-App Aftermarket services, including authorization and payment systems. And record evidence shows that developers (the buyers in the In-App Aftermarket of services in support of consummating a purchase of In-App Content) seek those services from third parties,[313] consistent with the notion of a *separate demand* from the matchmaking service offered via the app store in the Android App Distribution Market. The developer's demand for those services is derived from the consumer's demand for In-App Content. That numerous major developers have gone to the effort to use other systems, and to steer users to those systems (see Part IV.C.1 below), provides economic evidence that there is a market demand for the In-App Aftermarket services separate from the matchmaking services provided in the Android App

---

310. *Id*. at 383:3-21.

311. Google Developer Program Policy (effective Dec. 1, 2021), *available at* support.google.com/googleplay/android-developer/answer/11498144?hl=en&visit_id=637814760589469507-2803788482&rd=1. *See also* my discussion earlier in Part I.A.2.

312. *See* Google Play, *Google Play has something for everyone, available at* play.google.com/about/howplayworks/?section=about-google-play&content=service-fee. The service fee applies if developers "sell subscriptions or other digital content within an app," but is not affected by the length of time after an App is downloaded – with the exception of subscription products, which face a lower service fee after being retained for over a year. *See* Play Console Help, *Service fees*, *available at* support.google.com/googleplay/android-developer/answer/112622?hl=en.

313. *See, e.g.,* Koh Dep. at 183:14-18 ("There were developers that aspired to be -- have owned the end-to-end cycle. And yes, working with Google Play wouldn't allow developers to – developers aspired to own that, to own that, complete end to end."); Deposition of Kevin Wang (Dec. 15, 2021) at 94:14-21 (recalling that some developers were not interested in using Google's billing system). *See also* GOOG-PLAY-000259276 ("[redacted] is definitely not the only developer who would like to use their own payments system. There are only a handful who can build out a global system like ours, but that's not the only reason you'd want to use your own."); GOOG-PLAY-004703579 at GOOG-PLAY-004703584 (Google knows Epic, and [redacted] want to use their own billing, suspects [redacted] will as well); GOOG-PLAY-004716632 ("Google Play has repeatedly attempted to bring [redacted] on to our Billing Platform, but they have resisted citing poor results from testing GPB (two+ years ago) and a lack of *requirement* to use our platform due to some fuzzy language in our Play policy. (Many developers, including [redacted], interpret a loophole regarding downloadable content that may be used 'outside the app' to mean that they do not have to use Google Play Billing…)"); GOOG-PLAY-004721177 ([redacted] partners who are negotiating… are asking for carve outs to the Play Billing exclusivity requirement…"); GOOG-PLAY-006817773.R at GOOG-PLAY-006817802.R (Google keeps track of developers' many issues with Google Play Billing exclusivity).

Distribution Market.[314] Potential competitors include the major payment systems now used for online Internet purchases, such as credit and debit card networks, PayPal, distribution and payment systems used by other app stores, or developers themselves. I understand that Professor Schmidt finds that the software required for authorization and payments is already widely accessible and used for many online transactions and could readily be adapted to effectuate the purchase of Android In-App Content. Google froze out competitors by requiring that all developers use Google Play Billing, Google's own services for the fulfillment and consummation of transactions for all In-App Content purchased in Apps that were downloaded from the Play Store. Notably, in discussions with [redacted] Google had considered a scenario in which [redacted][315]

139. Google's prohibition of developer steering of consumers to outside authorization and payment channels indicates that third-party providers could replicate Google's authorization and payment channel for In-App Content, and that there is a separate demand for those services.[316] Google's internal documents recognize that developers sometimes demand, and utilize, alternative methods to accept payment for purchases of In-App Content.[317] Netflix, as one example, "views billing as a competitive advantage."[318]

140. Multiple internal analyses conducted by Google evidence that it considered [redacted]. One 2019 presentation contemplates that Google could [redacted][319] Likewise, a 2021 Google analysis titled "GPB as Platform of Choice" highlighted the ways Google could differentiate its offerings of the In-App Aftermarket such that Google could "elevate the product."[320] Google's internal analyses of competition among Google Play Billing and alternatives demonstrate [redacted] These competitive analyses further demonstrate that distinct services are offered in the In-App Aftermarket as compared to the Android App Distribution Market.

314. Although Google may have security justifications for approving distribution methods of In-App Content, this can easily and less restrictively be accomplished by white-listing developers, a process that Google already conducts for OEMs, and by providing developers with certificates approving their security so that developers can use them with other payment processors.

315. GOOG-PLAY-005653612.R at GOOG-PLAY-005653617.R ("Billing Integration for Android Market.").

316. Google uses the Google Play Billing Library API to facilitate the selling of In-App Content. *See* Google Play Billing, *available at* developer.android.com/distribute/play-billing.

317. Various developers have indicated a desire to use their own transaction services, declining the use Google Play Billing. [redacted] and others have invested in their own platforms and have expressed a desire to use a product other than Google's. See, e.g., GOOG-PLAY-000258923 (July 2018 email: "[redacted] does not believe there's a path forward on GPB at any rational terms given 1) the higher observed churn on our platform and 2) the strategic importance they place on payments."); GOOG-PLAY-000840773 at GOOG-PLAY-000840774 ("The larger (most profitable) services like match, Zoosk, and eHarmony, began as web businesses … and are actually quite sophisticated in terms of churn-reduction, reengagement, and buyer conversion. In some ways our billing platform is not as good as theirs currently are"); GOOG-PLAY-003334312 at -316 (noting developers have said "We want to give users choice of payment" and "my billing platform is a competitive advantage and would perform better for me"); GOOG-PLAY-004470512 at GOOG-PLAY-004470513 (email thread summarizing "[redacted] test (and their disappointing results)" from testing integration of Google Play Billing.

318. GOOG-PLAY-000259276 at GOOG-PLAY-000259277.

319. GOOG-PLAY-000542516.R at GOOG-PLAY-000542532.R (noting that GPB could provide "a native, secure payment solution for physical goods" as well as digital goods).

320. GOOG-PLAY-007745829 (further noting that "As Developers are coming onto GPB (post payments policy), some are noticing that GPB doesn't compare to previous solutions in performance A/B tests.").

141. Further, Google's documents recognize the possibility that payment systems for In-App Content could turn into full-blown competitive app stores. A Google document noted that "preventing payment circumvention" from in-app purchases could reduce the possibility that a nascent competitor in the In-App Aftermarket would expand to be a full-bore app store competitor in the adjacent Android App Distribution Market.[321]

142. Google's efforts to prevent selected developers from circumventing Google Play Billing and using alternative means of authorization and payment processing for In-App Content is yet additional evidence that there are actual or potential competitors to Google's services for these items, and thus that there is a separate and distinct In-App Aftermarket. Google's documents also provide evidence that, without its contractual restrictions, Google's take rate for In-App Content would fall. For example, Google modeled the downward pressure on its take rate if it were to allow selected developers to use lower-cost alternative means of payment processing and to steer users to those alternatives.[322] This exercise demonstrates that Google recognized that authorization and payment processing for In-App Content is distinct from what it provides through the Play Store.

B. The Relevant Geographic In-App Aftermarket Is Global (Excluding China)

143. The geographic market for the In-App Aftermarket is also global (excluding China). If Google's restrictions were not in place, it would not require any increase in Google's current take rates to attract entrants from other countries into the In-App Aftermarket. [redacted] Thus, the geographic In-App Aftermarket is not limited solely to the United States but is worldwide (excluding China).

C. Google's Market Power in the In-App Aftermarket

144. As with the Android App Distribution Market, I assess Google's power in the In-App Aftermarket through direct proof of firm-specific measures that speak directly to a firm's ability to profitably raise prices.[324] In Part V.C below, I show that Google would be forced to reduce its take rate in the In-App Aftermarket absent its restraints, which is also direct evidence of

---

321. GOOG-PLAY-004564758. Google expressed concern that competitive "[payment] services aggressively push at the boundaries of the Play policy. The most overt example of this is Amazon's push into app distribution, but it is now more commonly seen in the sale of virtual, cross-app currencies or in-app items through web or offline channels. The challenges that these services pose to Google Play is proportional to the degree to which any of these services establishes themselves in any given market." *Id.* at GOOG-PLAY-004564761. One of the "vectors of concern" was "payment disintermediation." *Id.* This is a similar anticompetitive theory of harm as that pursued by the Department of Justice (DOJ) in the *Microsoft* antitrust litigation, in which the dominant operating system tied its browser to its operating system with the alleged aim of preventing a rival browser from evolving into a full-bore competitor for Microsoft's operating system (or what the DOJ called "middleware").

322. GOOG-PLAY-006829073 [redacted]

323. EPIC_GOOGLE_00108372 (proposing [redacted] Other international in-app payment processors include Judopay (www.judopay.com/about), Paysafe (www.paysafe.com/us-en/), and Adyen (www.adyen.com/about)).

324. Herbert Hovenkamp, *Digital Cluster Markets*, COLUMBIA BUSINESS LAW REVIEW 1-30 (forthcoming 2022), at 24 ("By contrast, 'direct' proof relies on estimates of firm elasticity of demand, evidenced mainly by a firm's price-cost margins or output responses to price changes.[] These methodologies are capable of giving more accurate measures of market power as it is best defined, which is the ability of a firm to profit by raising its price above its costs.[]"); *see also* Louis Kaplow, *Why (Ever) Define Markets?,* 124 HARVARD LAW REVIEW 437 (2010).

its market power. That Google is able to impose restrictions on developers that exclude competition also evidences Google's power. I also examine indirect evidence—market shares and entry barriers—to establish that Google has market power in the In-App Aftermarket. My assessment of both direct and indirect evidence relies entirely on data and methods that are common to the Class.

1. Direct Evidence

145. Google's 30 percent standard take rate for initial downloads and purchases of In-App Content is direct evidence of its market power in the In-App Aftermarket. In the Android App Distribution Market, the Play Store brings together consumers and developers in a matchmaking role, though in many cases the consumer would be aware of the App she sought before searching for it in the Play Store, suggesting a more modest contribution by Google to value added.[325] In contrast, there is no matchmaking role in the purchase of In-App Content or for the services in support of consummating those purchases in the In-App Aftermarket.

146. Because Google as an intermediary provides additional value in the Android App Distribution Market via matchmaking, one would expect that, as in competitive markets where the take rate reflects the value added by the intermediary, the take rate in the Android App Distribution Market would exceed any take rate in the In-App Aftermarket. That Google's take rate—a price that presumably reflects its value-added to the transaction—is the same for initial downloads and purchases of In-App Content made even five years later strongly suggests that Google has anticompetitively extended its market power from the Android App Distribution Market into the In-App Aftermarket.

147. Indeed, a Google document shows that its elevation of take rates over competitive levels is substantially profitable, suggesting that Google possesses market power. In a 2020 business strategy document recommending reductions in the take rate to 15 percent one year after download (in other words, in the In-App Aftermarket), Google conceded "[t]here is a lot of external and internal pressure on our GPB [Google Play Billing] business model,"[326] suggesting that Google's artificial insertion of its billing system into the In-App Aftermarket at a 30 percent take rate was drawing ire from developers.[327] Google estimated that reducing the take rate to 15 percent after one year would reduce its revenues by roughly [redacted][328] implying that imposing the current 30 percent tax in perpetuity rather than 15 percent increases its profit for one year by [redacted]

325. Google's documents provide evidence that the majority of downloads from the Play Store are the result of users that "already know the app prior to the Store visit." GOOG-PLAY-007317574 at GOOG-PLAY-007317585. A 2020 presentation reviews survey results showing that "[redacted] of acquisitions are from [redacted]," while only "[redacted] of acquisitions are [redacted]." *Id.* at GOOG-PLAY-007317578. Similarly, when asked whether they knew about an acquired App before vising the Play Store, [redacted] percent responded affirmatively. *Id.* at GOOG-PLAY-007317584. Only [redacted] percent of users thought that information provided by the Play Store about an App (e.g., screen shots, number of downloads) was useful in making the initial installation decision. *Id.* at GOOG-PLAY-007317591.

326. GOOG-PLAY-006990552 at GOOG-PLAY-006990553.

327. *Id.* ("We recommend changing to a 30/15 service fee for (year 1 subsequent) with the optional addition of a distribution services revenue stream which will compensate for some kind of lost revenue and improve diversity but will also incite frustration in the set of developers who had previously paid nothing.").

328. *Id.* ("[redacted] after year 1 will cost [redacted]).

148. Google's ability to price discriminate also evidences its market power. Basic economic principles tell us that "[f]or a firm to price discriminate, it must have some market power."[329] In a document titled "[redacted]," Google recognized that its 30 percent take rate did not work for all developers, including media and streaming Apps.[330] Google even noted that its 30 percent take rate was "untenable" for certain Apps.[331] Project Hug resulted in effective take rate reductions for the largest developers, including in the In-App Aftermarket. Google also implemented programs for other developers such as the "Living Room Accelerator Program" ("LRAP") which lowered the take rate for Apps that delivered media-related In-App Content to users,[332] offering some developers a 15 percent take rate.[333] Some developers were offered even lower take rates: According to one document, "[redacted] is only one of a select group of developers that have been offered a [redacted] rev share."[334] Google has also cut in half the take rate for subscriptions, which are not limited to mobile devices; for subscriptions the platform is not as important, and they could command a lower rate. That Google is able to cut special deals below its standard 30 percent take rate for developers with recurring subscriptions or with streaming Apps with large content costs[335] is also consistent with Google having market power. A firm that lacks market power would not enjoy the privilege of discriminating across customer types according to their willingness or ability to pay, but instead would be forced to charge a uniform price at competitive levels.

2. Indirect Evidence

149. Ninety-seven percent of all developers offering apps through the Play Store currently use Google Play Billing, and a September 2020 announcement by Google indicates that it intended to capture the remaining three percent.[336] Google announced that it would fully enforce its contractual restrictions that require developers to use only Google Play Billing for all purchases of In-App Content. As a result, developers cannot use their own methods of authorization and payment processing services or contract for them through third parties. Alternative providers are

329. N. GREGORY MANKIW, PRINCIPLES OF MICROECONOMICS 303-304 (Cengage Learning 8th ed. 2018) [hereafter MANKIW] ("price discrimination is not possible when a good is sold in a competitive market"; "For a firm to price discriminate, it must have some market power.").

330. GOOG-PLAY-007329063.

331. *Id.* at GOOG-PLAY-0079066.

332. *See* Defendants' Answers and Objections to Developer Plaintiffs' First Set of Interrogatories to Defendants (July 6, 2021) at 13-16. *See also* Google Play Console, *Play Media Experience Program*, available at play.google.com/console/about/mediaprogram/.

333. GOOG-PLAY-000259276 at GOOG-PLAY-000259277 ("Conversations are ongoing regarding [redacted] enabling Google Play Billing for in-app purchases on a subset of Android TV OEMs…Exact economics still need to be figured out but could be close to LRAP's [redacted]).

334. GOOG-PLAY-006381385 at GOOG-PLAY-006381387.

335. *See, e.g.,* Kif Leswig, *Bumble, Duolingo lead rally in shares of app developers after Google slashes subscription fees*, CNBC (Oct. 21, 2021), *available at* www.cnbc.com/2021/10/21/google-cuts-app-store-fees-lifting-shares-of-app-developers.html.

336. Google - Android Developers Blog, *Listening to Developer Feedback to Improve Google Play* (Sept. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html ("Again, this isn't new. This has always been the intention of this long-standing policy and this clarification will not affect the vast majority of developers with apps on Google Play. Less than 3% of developers with apps on Play sold digital goods over the last 12 months, and of this 3%, the vast majority (nearly 97%) already use Google Play's billing. But for those who already have an app on Google Play that requires technical work to integrate our billing system, we do not want to unduly disrupt their roadmaps and are giving a year (until September 30, 2021) to complete any needed updates. And of course we will require Google's apps that do not already use Google Play's billing system to make the necessary updates as well.").

totally foreclosed from the In-App Aftermarket. Google's requirement that developers of Apps marketed through the Play Store in the Android App Distribution Market exclusively use Google Play Billing for all purchases of In-App Content serves as an artificial barrier to entry in the In-App Aftermarket. Absent the restraint, many providers would step forward to provide competitive payment processing and authorization of In-App Content.

D. Google's Anticompetitive Exclusionary Conduct in the In-App Aftermarket

150. Google has maintained multiple restrictions affecting the In-App Aftermarket. These fall into three mutually reinforcing categories. By contract, Google conditions the right to distribute an App downloaded through the Play Store on a developer's agreement to exclusively use Google Play Billing for all subsequent sales of In-App Content.[337] Google contractually requires developers to pay Google a set take rate (generally at 30 percent) for every purchase of In-App Content made through their Apps in perpetuity. Put differently, Google enforces this condition by requiring the developer to use Google Play Billing for all payments of In-App Content within the App forever and at the same take rate it commands in the Android App Distribution Market.[338] If a consumer downloads an App via the Play Store, the developer is charged a commission for any purchases made within the App, even long after the Play Store performed its initial matchmaking and distribution function. Thus, Google has extended its monopoly power in the Android App Distribution Market to insert itself into the separate In-App Aftermarket by requiring developers to exclusively use Google Play Billing for authorization and payment services in support of the purchase of In-App Content and to pay Google a take rate on those purchases (usually 30 percent), which can economically be characterized as a "tie-in."

151. Google also contractually prohibits developers from steering customers to alternative authorization and payment processing outlets for purchasing In-App Content outside the Play Store, including the developer's web site or alternative suppliers of payment processing and other services in the In-App Aftermarket.[339] And Google even prohibits the developer from using any consumer information learned through the Play Store. These restraints constrain an App's steering to lower-cost alternatives via browser-based payment options such as "in-app web views, buttons, links, messaging, advertisements, or other calls to action."[340]

152. In contrast, other app stores allow developers the ability to select their providers of payment systems for purchases of In-App Content at lower take rates than Google imposes. For example, Aptoide imposes a ten percent take rate for purchases of In-App Content if the user

---

337. Google – Play Console Help, *Developer Program Policy* (effective December 1, 2021), *available at* support.google.com/googleplay/android-developer/answer/11498144?hl=en&visit_id=637814760589469507-2803788482&rd=1. *See also* my discussion earlier in Part I.A.2.

338. *See* Google - Play Console Help, *Service fees*, *available at* support.google.com/googleplay/android-developer/answer/112622?hl=en. The service fee applies if developers sell subscriptions or other digital content within an app, but is not affected by the length of time after an app is downloaded – with the exception of subscription products, which face a lower service fee after being retained for over a year.

339. Google Developer Program Policy (effective March 1, 2021), *available at* support.google.com/googleplay/android-developer/answer/9934569?hl=en&ref_topic=9877065#zippy=%2Cmarch; *see also* GOOG-PLAY-000225435 ("it's against policy to direct users to content outside of the Play Store (including to 3rd party websites offering rewards").

340. *Id.*

downloads the app using the developer's own URL.[341] The One Store in South Korea imposes a five percent commission for developers that do not use the One Store billing system.[342]

1. Google's Contractual Provisions with Developers Enable Google To Maintain Its Dominance in the In-App Aftermarket

153. In other settings, the long-term or perpetual arrangements that Google has imposed on developers might have been difficult to enforce. That is not the case here. By requiring developers to use Google Play Billing in support of the purchase for all In-App Content, Google can readily monitor and enforce its take rates, enabling extraction of a supra-competitive commission for as long as the App is used and In-App Content is purchased.

154. Only about three percent of the developers that sold In-App Content in the In-App Aftermarket through Apps initially downloaded from the Play Store in 2020 were able to circumvent the restriction, although Google later announced that it was going to enforce the restriction on this group.[343] Those few developers able to circumvent Google's restrictions have garnered a critical mass of consumers and widespread name recognition, which has enabled them to allow users to purchase their In-App Content from other platforms. The most prominent developers in this category are Netflix, Amazon, Spotify, Match Group, and Epic.[344] A company like Netflix, a household name with over 200 million subscribers, does not want to hand over to Google 30 percent of first-year subscriber revenue simply because a consumer created her account on her Android App. Circumventors like Netflix were able to steer users to their websites, where consumers could create a new account not subject to Google Play Billing.[345] Developers engaging in this steering had to do so indirectly—through communications outside the App—because Google's agreements with developers preclude them from explicitly steering users to the

341. Revenue share, *available at* docs.catappult.io/docs/distribution-and-revenue-share ("Self-distribution opens the possibility for the developer to promote and distribute their apps through their own channels and earn a revenue share of 90%").

342. GOOG-PLAY-000005203 at GOOG-PLAY-000005221; GOOG-PLAY-000005264. *See also* Kim Byung-wook, *Google's app billing plan continues to backfire*, THE KOREA HERALD (June 28, 2021), *available at* www.koreaherald.com/view.php?ud=20210628000824 ("Unlike Google, ONE store allows app developers to operate their own billing systems. In this case, the cut is 5 percent.").

343. Google - Android Developers Blog, *Listening to Developer Feedback to Improve Google Play (*Sept. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html

344. EPIC_GOOGLE_00123016; EPIC_GOOGLE_01389946. Google still holds significant power over these companies. For instance, Match Group's chief legal officer, Jared Sine, in testimony before the United States Senate detailed communications from Google inquiring into why his testimony might differ from what they had already discussed. "When you receive something like that, Senator, from a company that can turn you off overnight, you're always a little intimidated," said Sine. He added, "We're all afraid, is the reality, Senator." Angel Au-Yeung, *App Providers Are 'All Afraid' Of Apple's And Google's Market Power, Match Group And Spotify Tell Senate*, FORBES (Apr. 21, 2021), *available at* www.forbes.com/sites/angelauyeung/2021/04/21/app-providers-are-all-afraid-of-apples-and-googles-market-power-match-group-and-spotify-tell-senate/?sh=4a622ae9596c. *See also* GOOG-PLAY-000559379 at GOOG-PLAY-000559382 ("We have found that, in gaming, the concerns over revenue share predominantly come from the largest players who have built significant businesses and are now focused on improving margins.").

345. Tinder also accomplished steering by means of differential pricing—offering significantly lower prices for its subscription services when users signed up through its website. EPIC_GOOGLE_02075797 ("The Tinder app charges $29.99 a month for a Gold membership (which shows you everyone who's swiped right on you). Tinder's website charges just $13.49 a month for the same service.").

developer's website for digital content purchases.[346] The vast majority of developers do not possess the requisite widespread user adoption, name recognition, and clout to circumvent Google's restriction.

2. Google's Revenue-Sharing Agreements with Developers Have Substantially Eliminated the Threat of Defection to Alternatives in the In-App Aftermarket

155. Google has taken steps to rein in any developers whom it perceived might threaten to use alternative authorization and billing systems in the In-App Aftermarket to avoid Google's take rate.

156. Initially, Google limited Project Hug (described in Part III.D.3 above) to the [redacted] game developers—ostensibly to provide "increased value to both Google and developers while easing developer agitation about Google Play's revenue share."[347] Google chose these developers because [redacted][348] Indeed, Google estimated that defection by these developers could cost Google "~ [redacted] *margin* impact annually in three years [redacted] revenue / [redacted] margin risk cumulative 2019-2022)."[349] Google estimated the cost of risk mitigation through Project Hug payments at only [redacted] through 2022.[350] Although the primary purpose of Project Hug was to prevent developers from supporting a rival app store, a secondary purpose was to prevent an alternative form of payment processing in the In-App Aftermarket from taking root.

157. Depositions to date have confirmed the success of Project Hug in foreclosing competition. When one of the key employees at Google in charge of Project Hug left Google in 2020, he assessed that "Project Hug was working," and he "was not aware of any Project Hug developer pursuing a new Android distribution platform."[351]

158. Google also introduced a variety of "Velocity" programs, aimed at inducing other powerful developers to comply with its Google Play Billing restrictions, including: (1) the LRAP, which focused on live TV and video Apps; (2) the Audio Developer Accelerator Program

---

346. When Epic explicitly steered consumers to their own payment system rather than Google's system by providing a 20% discount, Google removed Epic's Fortnite from the Play Store. *See* Jack Nicas, *How Apple's 30% App Store Cut Became a Boon and a Headache*, NEW YORK TIMES (Aug. 14, 2020), *available at* www.nytimes.com/2020/08/14/technology/apple-app-store-epic-games-fortnite.html. Google states that developers are allowed to use other platforms in addition to Google Play, but that they cannot directly advertise alternative options through their App. *See* Google - Play Console Help, *Understanding Google Play's Payments policy*, *available at* support.google.com/googleplay/android-developer/answer/10281818?hl=en#zippy=%2Ccan-i-distribute-my-app-via-other-android-app-stores-or-through-my-website%2Ccan-i-communicate-with-my-users-about-alternative-ways-to-pay%2Ccan-i-communicate-with-my-users-about-promotions-on-other-platforms.

347. *See* GOOG-PLAY-000559379 at GOOG-PLAY-000559382 ("Thus far we have avoided altering our revenue share business model and instead have engaged this limited set of game developers [redacted] devs) in cross-Google commercial deals to both build deeper relationships with these partners and reduce agitation around our 30% revenue share."). *See also* Koh Dep. at 153:10-19 (Project Hug targeted developers that Google projected would constitute [redacted] of overall Play Store consumer spend in 2019) (testifying about PX. 136 – GOOG-PLAY-003332817.R at GOOG-PLAY-003332822.R).

348. GOOG-PLAY-000233314

349. GOOG-PLAY-000000807

350. *See* GOOG-PLAY-000004762.R at GOOG-PLAY-000004764.R.

351. Koh Dep. at 368:4-369:4.

("ADAP"), which focused on music Apps; and (3) the App Velocity Program ("AVP"), a generalized Project Hug-for-Apps program for high-priority partners that did not fit into any of the other categories.[352] While the ostensible purpose of the LRAP and ADAP programs was to accommodate "constrained dev margins," the AVP's goal was, like Project Hug, to "boost integration with Play Billing" by offering to "reinvest ~10% in-kind."[353]

E. Google Cannot Claim That Its Supracompetitive Profits Are Constrained by the "Single Monopoly Profit" Theory

159. In the 1970s and 1980s, economists belonging to the "Chicago School" of economics—which promotes the virtues of free-market principles[354]—condoned many exclusionary strategies with an economic theory that has become known as the "single monopoly profit" ("SMP") theory. The SMP theory posits that, since a monopolist will always find a way to fully exploit its monopoly profit in the market it has monopolized, regardless of the existence of a secondary market or aftermarket, any exclusionary conduct in the secondary market is motivated by procompetitive reasons, such as vertical integration efficiencies, and should not be condemned as anticompetitive.[355]

160. Applied to this case, Google may erroneously argue that the SMP theory could be interpreted to deem Google's Aftermarket Restrictions procompetitive and lacking in harm to competition or consumers. It would posit that if Google were prohibited from engaging in the Challenged Conduct, Google would merely raise its take rate in the Android App Distribution Market or impose some other fees to fully restore the profits it now extracts from the In-App Aftermarket. SMP theory suggests that, so long as Google had a monopoly in the Android App Distribution Market, it would find a way to fully extract a monopoly profit from that market, such that there would be no incentive to further monopolize the In-App Aftermarket. In this section, I explain how two of the key assumptions that undergird the SMP theory are not satisfied, meaning that the profits Google has extracted from the In-App Aftermarket were not available to Google solely on the basis of its monopoly in the Android App Distribution Market. Accordingly, Google's anticompetitive conduct in the In-App Aftermarket was motivated to extract incremental supra-competitive monopoly profit from developers.

161. A wave of new economic research in the 1990s and 2000s has shown that the implications of the SMP theory hold only if certain assumptions that underlie it also are true.[356] As it is applied to artificially linking a monopolized service—here, Android App distribution—with a product or service in another market—here, services in support of the purchase of In-App Content—the five conditions under which the SMP theory holds are:

---

352. GOOG-PLAY-004144047.R at GOOG-PLAY-004144052.R (explaining this program would only be available to those developers who represented [redacted] (or more) of total Play Spend).

353. GOOG-PLAY-003881390.R.

354. Akhilesh Ganti, *Chicago School of Economics*, INVESTOPEDIA (Feb. 21, 2021), *available at* www.investopedia.com/terms/c/chicago_school.asp.

355. *See, e.g.*, Robert Bork, THE ANTITRUST PARADOX: A POLICY AT WAR WITH ITSELF (Bork Publishing 2021); Aaron Director & Edward H. Levi, *Law and the Future Trade Regulation,* 51 NW. U. L. REV. 281 (1956).

356. For a review of the economic literature, see Einer Elhauge, *Tying, Bundled Discounts and the Death of Single Monopoly Profit Theory*, 123(2) HARVARD LAW REV. 397-481 (2009).

(1) buyers do not use varying amounts of the linked product or service (in the In-App Aftermarket);

(2) buyers exhibit a strong positive correlation in their demands for the linked product or service (in the In-App Aftermarket) and the linking products or services (in the Android App Distribution Market);

(3) buyers do not use varying amounts of the linking product or service (Android App distribution);

(4) the competitiveness of the linked market (the In-App Aftermarket) is fixed; and

(5) the competitiveness of the linking market (the Android App Distribution Market) is fixed.[357]

162. When any one of these assumptions is not supported empirically, then the pro-competitive justifications of SMP theory do not hold.[358] Here, assumptions (1) and (4) are not satisfied. Failure to satisfy either one would undermine application of SMP theory. Failure to satisfy both assumptions strengthens that conclusion. Because SMP theory does not apply here, Google is using its Aftermarket Restrictions to extract profits it otherwise would not be able to obtain. The inapplicability of SMP theory implies an anticompetitive motivation for Google's Aftermarket Restrictions.

163. Recall that Google's take rate in the In-App Aftermarket is tethered to the spending on In-App Content by the consumer. Although the buyer in the In-App Market is the developer, the demand for the services in support of consummating the purchase of In-App Content is derived from the demand for the In-App Content itself. Applied to this case, for Google to extract consumer surplus solely by virtue of its market power in the Android App Distribution Market, assumption (1) requires that consumers do not purchase varying amounts of In-App Content. This is plainly false. Different consumers purchase different quantities of In-App Content, and therefore Google's revenues through its take rates anticompetitively applied to those purchases vary by consumer. This is evident from my examination of Google's transaction database. Figures 4, 5, and 6 show the distributions of purchases of In-App Content for three of the top-selling games.

357. *Id*. at 404 ("However, the model indicating a single monopoly profit *depended on several key assumptions*: (1) fixed usage of the tied product; (2) strong positive demand correlation; (3) fixed usage of the tying product; (4) fixed tied market competitiveness; and (5) fixed tying market competitiveness. As the economic literature shows, different results are reached if one relaxes these narrow assumptions. Indeed, relaxation of any one of these assumptions produces a distinctive profit increasing effect.") (emphasis added).

358. *Id*. For example, when assumption (1) is violated and consumers use varying amounts of the linked product or service, tying can be used to extract consumer surplus, with buyers who use more of the tied product effectively paying more for the same product. Discriminating with ties may be more effective than traditional price discrimination—that is, charging a different price to each buyer—if the firm could not otherwise tell how much buyers value the tying product.

[redacted]

[redacted]

164. As the figures demonstrate, the consumers' purchases of In-App Content vary widely across buyers, proving that assumption (1) is violated. The varying amounts of purchases of In-App Content provides an anticompetitive motivation for Google's extension of its monopoly power in the Android App Distribution Market into the In-App Aftermarket to extract more revenue from its monopoly position—an opportunity that would not exist in the absence of the Aftermarket Restrictions. Google may argue that it inserted itself into the In-App Aftermarket via the restraint requiring developers to use Google Play Billing for all transactions so as to monitor and "meter" both a customer's usage and a developer's in-app sales for an App. However, as a matter of economics, Google cannot extract the same level of monopoly profits through this restriction as it would solely through a monopoly in the Android App Distribution Market. Given the wide range of purchases of In-App Content, it would be very difficult for Google to predict, *ex ante*, a given consumer's propensity to make particular in-app purchases. Such a prediction would be needed by Google to replicate a metering strategy with upfront pricing for Apps sold solely in the Android App Distribution Market through the Play Store.[359] The problem for Google is that it is impossible to know *ex ante* how much any given consumer/developer combination will use an

359. A "metering tie" occurs whenever a firm meters usage of its product by requiring users to purchase at elevated prices a tied product that is needed to use its product, as in the case of printers and ink. Economists have shown that, contrary to the Chicago School, metering ties are often used for anticompetitive purposes. *See, e.g.*, Einer Elhauge & Barry Nalebuff, *The Welfare Effects of Metering Ties*, 33 JOURNAL OF LAW, ECONOMICS & ORGANIZATION 68 (2017) (showing that metering ties (1) *always* lower consumer welfare and total welfare unless they increase capital good output, and (2) *always* harm consumer welfare, even when output increases, under realistic market conditions in which usage rates are independently distributed from per-usage valuations). Applied here, there is no credible theory under which Google's restrictions increased output in the Android App Distribution Market. Indeed, to the extent a foreclosed rival in the In-App Aftermarket could have evolved into a competitive app store, Google's restrictions could have reduced output in the Android App Distribution Market as well. *See, e.g.*, GOOG-PLAY-004564758 (noting that "preventing payment circumvention" from in-app purchases could reduce the possibility that a nascent competitor in the In-App Aftermarket would expand to be a full-bore app store competitor in the adjacent Android App Distribution Market).

App and subsequently generate any in-app purchases so as to be able to price Apps to generate a supra-competitive monopoly profit for Google.

165. In addition to assumption (1) being violated, assumption (4) requires that the "competitiveness" of the In-App Aftermarket be "fixed"—that is, the tied market is perfectly competitive in a way that tying cannot alter. Applied here, that would mean that In-App Aftermarket rivals (such as independent payment processors or developers) face no entry or fixed costs and can expand instantaneously to supply the whole market.[360] If true, then the extent of competition in the In-App Aftermarket would be invariant to Google's tie-in, in the sense that Google's inserting its payment processing service into the In-App Aftermarket cannot be used to reduce the competitiveness or efficiency of rivals or potential rivals in the In-App Aftermarket.[361] This assumption is not true, however, because Google's contractual linkage is specifically designed to prohibit developers from using any alternative to Google Play Billing to provide In-App Aftermarket services. In the absence of the requirement, there would be myriad third-party providers of payment processing and other In-App functions for developers (such as authorizing the use of In-App Content, record keeping, and server hosting), which would engender competition on take rates.

166. Moreover, the infrastructure to provide payment processing (and authorization) exhibits economies of scale;[362] the same payment system (or record-keeping system or server) can be used for each additional transaction, with little additional marginal cost. Accordingly, depriving third-party competitors in the In-App Aftermarket of scale economies makes them weaker competitors and alters the state of competitiveness in the In-App Aftermarket. Google's tie-in can foreclose enough of the tied market to make entry revenues, spread across a lower base of transactions, lower than entry costs. Just as a monopoly resort that requires guests to eat all meals on the property (a tie-in) can foreclose independent local restaurants and prevent them from achieving the requisite economies of scale,[363] so too has Google foreclosed rival suppliers in the In-App Aftermarket, resulting in the ability for Google to charge supra-competitive take rates on purchases of In-App Content. In an open, competitive In-App Aftermarket, consumers would be free to choose an alternative payment processor, and take rates in the In-App Aftermarket would fall towards competitive levels. In the next section, I offer two models that can be used to estimate

---

360. Einer Elhauge, *Tying, Bundled Discounts, and the Death of the Single Monopoly Profit Theory*, 123(2) HARVARD LAW REVIEW 397-481, 413 (2009).

361. *Id.* ("the economic literature summarized below shows that a tie that forecloses enough of the tied market can reduce rival competitiveness by impairing rival efficiency, entry, existence, aggressiveness, or expandability.").

362. *See, e.g.*, Oxera, *Paying up: the new economics of payment systems* (Jun. 30, 2020), *available at* www.oxera.com/insights/agenda/articles/paying-up-the-new-economics-of-payment-systems/ ("Retail payments have long been characterised by the following three economic features …. economies of scale—which mean that it can be more efficient to operate a platform with a large number of users (although regulatory and technical developments are tending to reduce the importance of this)."); Risto Gogoski, *Payment systems in economy - present end future tendencies*, 44 PROCEDIA - SOCIAL AND BEHAVIORAL SCIENCES 436–445, 438 (2012) ("The payment industry also exhibits considerable economies of scale. First, the value that an individual participant derives from using a particular payment system increases with the number of other parties using that same system. Second, high levels of initial investment (called 'sunk costs') are required in order to establish a payment system, and considerable fixed costs are incurred in the operation of such a system (more payments less costs).").

363. This example comes from Dennis Carlton, *A General Analysis of Exclusionary Conduct and Refusal to Deal—Why* Aspen *and* Kodak *Are Misguided*, 68(3) ANTITRUST LAW JOURNAL 659-983 (2001).

those reduced take rates in the Android App Distribution Market and in the In-App Aftermarket, respectively.

V. The Challenged Conduct Can Be Shown To Have Generated Antitrust Impact Using Common Methods and Evidence

167. A key difference in the two relevant antitrust markets—the Android App Distribution Market being two-sided and the In-App Aftermarket being one-sided—allows for different methods for assessing impact or what take rates and consumer subsidies would prevail in each market without Google's Challenged Conduct. I discuss these two methods and their implications in turn. In both cases, the modeling inputs are common to the class. In Part V.B, using a two-sided model developed by Rochet and Tirole, where the locus of competition is on the developer take rate, I show classwide impact for those Class members in the Android App Distribution Market. In Part V.C., using a one-sided model of price competition, I show classwide impact for Class members in the In-App Aftermarket. In Part V.E, again using the two-sided model developed by Rochet and Tirole, where the locus of competition is instead on the consumer subsidy, I show classwide impact for those Class members. In Part VI, I estimate aggregated damages to the Class based on these impact models.

168. My analysis differs from the standard regression analysis commonly used in many price-fixing matters to isolate the effects of anticompetitive conduct in a limited timeframe compared to a competitive market absent the challenged restraints. Because Google has imposed the Challenged Conduct in both the Android App Distribution Market and the In-App Aftermarket since those markets were originally formed, there is no pre-existing or post-conduct time period to use for purposes of standard regression analysis. Accordingly, I employ widely accepted economic models to determine take rates that would be charged in a hypothetical but-for world without the Challenged Conduct. Before introducing the models, I briefly explain how multi-homing (by customers and developers) and steering (by developers) would put downward pressure on take rates in the absence of the Challenged Conduct.

A. Multi-Homing and Steering Would Put Downward Pressure on the Take Rate That Google Imposes on App Developers

169. As I will show here using evidence and methods common to the Class, Google's Challenged Conduct has enabled Google to charge developers supra-competitive take rates in the Android App Distribution Market and the In-App Aftermarket. With its dominance in the Android App Distribution Market and consequent market power in the In-App Aftermarket, Google is able to extract a supra-competitive take rate on all paid App downloads and purchases of In-App Content. This is true even after Google's reduction in its take rates announced in March 2021 (from 30 percent to 15 percent on the first $1 million of developer revenue)[364] and for subscription

364. Sameer Samat (Google Vice President, Product Management), *Boosting Developer Success on Google Play* (Mar. 16, 2021), *available at* android-developers.googleblog.com/2021/03/boosting-dev-success.html ("Starting on July 1, 2021 we are reducing the service fee Google Play receives when a developer sells digital goods or services to 15% for the first $1M (USD) of revenue every developer earns each year.") Although the new policy applies to all developers, the overall reduction in the take rate is less significant for larger developers, because it applies only to the first $1 million in revenue. For example, developer with $10 million in revenue would pay a 15 percent take rate on the first $1 million, and a 30 percent take rate on the remaining $9 million, which works out to an overall take rate of 28.5 percent.

payments after the first year as of January 1, 2018 (reduced to 15 percent).[365] The concepts of multi-homing and steering are critical to understanding how Google's contractual restraints with OEMs, developers, and mobile carriers work as an economic matter.

1. Multi-Homing

170. Multi-homing, as the name suggests, occurs whenever buyers or sellers on the opposite sides of a two-sided platform use more than one platform for the same or similar purpose. For example, many young Internet users have social media accounts on two or more social media platforms (e.g., Facebook and TikTok). Many ride-sharing drivers and riders have both Uber and Lyft on their phone, another form of multi-homing. Multi-homing is not exclusive to the digital world: People carry two or more credit cards in their wallets, and the stores they frequent accept more than one card, although anti-steering rules imposed by one of the most popular credit cards may inhibit card competition.[366]

171. In the context of this case, multi-homing exists to the extent consumers have app stores side-by-side on their mobile phone's home screens (if Google's conduct did not prevent consumers from having multiple app stores)—the adjacent placement is necessary so that multi-homing is equally convenient for consumers. When two platforms are sufficiently close substitutes in the eyes of buyers and sellers, multi-homing can lead to competitive outcomes that benefit both buyers and sellers.[367]

172. Multi-homing would occur absent the Challenged Conduct, as developers would be willing to distribute their applications through alternative app stores if they could achieve sufficient reach by doing so. And users would be willing to install the second app store on their home screens if (1) they could access their favorite Apps on a rival app store, and (2) if at least some of those Apps were available at a lower price on the second app store—a phenomenon that, in a competitive world absent Google's restrictions, would be made possible via steering.[368]

---

365. Although Google decreased the take rate for subscription services from 30 percent to 15 percent in 2018, the decrease was only applicable after the first year. (It wasn't until January 1, 2022, that the take rate was reduced to 15 percent for all subscription services across the board). Google documents indicate that [redacted] percent of consumer spend on in-app purchases occurs within one year of the consumer's first purchase, which would have limited the effect of the 2018 policy. *See* GOOG-PLAY-007819776 at GOOG-PLAY-007819909. Moreover, Google estimates in the lead-up to the announcement found only a [redacted] revenue loss from the change, because only [redacted] percent of active subscriptions and [redacted] percent of revenue from subscriptions came after 12 months. *See* GOOG-PLAY-000446626 at GOOG-PLAY-000446629. Another Google analysis calculated the effective take rate resulting from the 2018 policy change at [redacted] percent. *See* GOOG-PLAY-001291233 at GOOG-PLAY-001291251. In contrast, the competitive but-for world is one in which all developers would have enjoyed substantial and permanent reductions in the take rate and would be characterized by long-run equilibrium price adjustments to substantially developer lower costs flowing from substantially lower take rates. As a consequence, I conclude that Google's reduction in the take rate for subscriptions is unlikely to provide an adequate natural experiment that could be used to accurately measure the extent to which consumers would have paid lower prices in the but-for world.

366. Of course even if there is multi-homing, restrictions can create less than a competitive outcome. *See, e.g.,* Kevin Caves & Hal Singer, *Competing Approaches to Antitrust: An Application in the Payment Card Industry*, 27(3) GEORGE MASON LAW REVIEW 823-861 (2020).

367. Susan Athey & Fiona Scott Morton, *Platform Annexation*, Stanford Inst. for Econ. Policy Research Working Paper 21-015 (March 2021).

368. Although my primary impact model focuses on price effects (over the take rate), it is possible that competition would occur on non-price quality dimensions as well. For example, a specialized app store could emerge that provided better discoverability features, forcing Google to compete on that dimension.

2. Steering

173. Steering can exist in any type of market, but in the context of the two-sided platform present in this case, steering would entail a developer charging differential prices to consumers based on which platform the consumer selects, from which to download an App. The developer's aim, if steering were allowed, would be to induce consumers to transact over a lower-cost platform.[369] Economists have shown that, in a platform setting, steering puts downward pressure on the prices charged by sellers (here, developers), and thus anti-steering restraints are almost certainly harmful to competition.[370] Indeed, Google itself was well aware of the power of steering to put downward pressure on take rates.[371]

174. Steering occurs regularly across platforms in other industries where there are no restrictions that prevent it. One example is the market for "daily deals"—or discounted prices on certain products or services on a specific day—offered by platform apps like Groupon and Living Social. Empirical research has shown that in markets where there is platform competition, sellers on the sites offer more valuable promotions to buyers at lower prices relative to markets without platform competition.[372]

175. With multi-homing and steering—both conditions are necessary for competition to drive down prices to consumers—developers could, and would be incentivized to, charge a lower price for Apps to consumers who download Apps from a lower-cost app store platform. Consider a scenario in which a developer faced two take rates: Google's 30 percent rate and a rival app store's 15 percent rate. For simplicity, assume the developer charges $1 for downloading the App. In a world with multi-homing, the developer would have strong incentives to steer its customers to the lower-cost platform, as doing so would save it $0.15 per download (equal to the product of the 15 percent differential in take rates and $1). Indeed, the developer would be willing to offer up to a $0.15 reduction in the price of the App to steer its customers to the lower-cost platform. As more customers shift their downloads to the rival platform, Google would be forced to revisit its take rate; a lower Google take rate would in turn induce developers to lower their prices on the Play Store. I model this competitive dynamic formally in the following sections.

---

369. *See, e.g.,* Rochet & Tirole at n. 3 ("The occurrence of steering is easiest to visualize in those illustrations in which platforms charge per-end-user-transaction fees: The seller of a house or a B2B supplier may only list the house or the wares on the cheapest platform.").

370. *See, e.g.,* Benjamin Edelman & Julian Wright, *Price Coherence and Excessive Intermediation,* 130 Q. J. ECON. 1283 (2015); Rong Ding, *Merchant Internalization Revisited,* 125 ECON. LETTERS 347 (2014); Rong Ding & Julian Wright, *Payment Card Interchange Fees and Price Discrimination*, 65 J. IND. ORG. 39 (2017). For an overview of the intersection of multi-homing and steering, see Erik Hovenkamp, *Platform Antitrust*, 44 JOURNAL OF CORP. LAW 713 (2019) ("A second type of steering is undertaken by sellers on one side of a transaction platform. In most situations where buyers and sellers both multi-home, the buyer ultimately chooses the platform used to mediate his transactions. … Alternatively, the seller may vary the prices it charges in transactions over different platforms, applying a surcharge to those it disfavors (or, equivalently, a discount for transactions on its preferred platform). Such steering efforts were forestalled by the restraint at issue in *AmEx*, which is discussed further below.").

371. GOOG-PLAY-006829073.R at GOOG-PLAY-006829085.R.

372. Kim et al., *Two-sided platform competition with multihoming agents: An empirical study on the daily deals market*, 41 INFORMATION ECON. AND POLICY 36-53 (2017).

B. A Two-Sided Platform Model with Multi-Homing Shows That Google Would Be Compelled to Lower Its Take Rate from Developers in the Absence of Google's Android App Distribution Market Restraints

176. I start by analyzing the impact of the Challenged Conduct in the Android App Distribution Market. For this purpose, I use a model based on the one developed by economists Jean-Charles Rochet and Jean Tirole (winner of the Nobel prize in economics for, among other things, his pioneering work on monopolized industries) who formalized the economic framework for two-sided markets.[373] This framework has been widely cited by other economists.[374]

177. The model shows by how much Google's take rate on paid initial App downloads in the Android App Distribution Market would fall if the locus of competition occurs on the developer side of the platform once Google's multiple restraints and technical barriers are removed. The model shows what Google would charge developers in the presence of multi-homing and steering, which would occur in the absence of Google's exclusionary restraints. Although Google's exclusionary conduct was aimed at myriad potential entrants—including mobile carriers, OEMs, and developers—my analysis of a potential but-for world requires entry by only one rival app store platform with a comparable number and quality of Apps as are available in the Play Store. Although Google has consistently charged a take rate at (or very close to) 30 percent for the vast majority of consumer expenditures,[375] my analysis also takes account of Google's current take rates. For example, the take rate has been lowered from 30 to 15 percent for (1) subscription App renewals beginning January 1, 2018, and (2) transactions made for Apps and In-App purchases for a developer's first $1 million in annual sales beginning July 1, 2021.[376] I discuss the basic intuition behind this model and show how it can be readily adapted to the current setting.

178. In the event that the factfinder concludes that the Android App Distribution Market and In-App Aftermarket are not two separate markets, I have performed two alternative analyses, both of which apply the two-sided market framework to a single, combined market. These analyses (presented Part V.E and Appendix 4) contemplate competitive scenarios in which platforms compete for all transactions (both initial downloads and in-App purchases) in the aggregate. In

---

373. Rochet & Tirole.

374. *See, e.g.*, Marc Rysman, *The Economics of Two-Sided Markets*, 23(3) JOURNAL OF ECONOMIC PERSPECTIVES 125-43 (2009); Avi Goldfarb and Catherine Tucker, *Digital Economics*, 57 JOURNAL OF ECONOMIC LITERATURE 3-43 (2019); Joseph Farrell & Paul Klemperer, *Coordination and lock-in: Competition with switching costs and network effects,* in MARK ARMSTRONG AND ROBERT PORTER EDS., 3 HANDBOOK OF INDUSTRIAL ORGANIZATION (Elsevier 2007).

375. *See, e.g.*, Table 5, *infra*, Row 3 (showing that Google collected service fees in excess of [redacted] percent of consumer expenditures from 8/16/2016 – 12/31/2020).

376. The relatively few developers who paid reduced take rates in the actual world would also have paid take rates below the but-for level. Google's anticompetitive conduct resulted in a substantially inflated headline take rate of 30 percent, which is economically equivalent to inflating the list price of a product in an antitrust context. Customers that receive discounts from an inflated list price still incur antitrust injury because the discounts they receive are tied to the list price. *See, e.g.,* Hal Singer and Robert Kulick, *Class Certification In Antitrust Cases: An Economic Framework,* GEORGE MASON LAW REVIEW (2010), 1046-47, 1049 (explaining that class members are impacted even when they receive discounts relative to an inflated list price; here the list price is Google's headline take rate of 30 percent). Thus, even developers who paid reduced take rates in the actual world would have also paid lower take rates in the but-for world, and would have passed on some of the resulting savings to consumers. In Part VII, I demonstrate how damages can be calculated for individual Class Members using common methods, taking into account the fact that a limited number of developers received discounts relative to Google's standard 30 percent take rate.

Appendix 4, the locus of competition occurs on the developer side of the platform. In Part V.E, it occurs on the consumer side.[377]

1. The Platform Model in a Monopolized Setting

179. The two sides of the Android App Distribution Market are consumers of initial App downloads (buyers) and App developers (sellers). Google sets the commission it has charged developers for using the Play Store. Google does not charge consumers for accessing the Play Store and instead offers a small subsidy in the form of its loyalty points program, Google Play Points, effectively implying a small *negative* price (or subsidy) for consumers using the Play Store.[378] Importantly, while Google sets the commission charged to developers, developers set their own prices on App downloads.[379] As my extension of the Rochet-Tirole model illustrates, developers that can offer their Apps on an app store that charges a lower commission than Google will be incentivized to "steer" consumers to the alternative app store by charging lower prices on downloads in that alternative app store than they charge in the Play Store. In this way, consumers would also benefit from competition between app stores.

180. I begin by outlining the classic two-sided market model in which a platform operator sets per-transaction platform prices on both sides of the market where the platform operator has a monopoly (the "foundational monopoly model"). I then demonstrate how this model is easily extended to the instant case, where Google sets a take rate or commission imposed directly on developers instead of a per-transaction price and provides a subsidy to consumers in the form of loyalty points (the "applied monopoly model"). A portion of the supracompetitive cost imposed on developers through the take rate is passed through to consumers (which I show in Section V.D below). I then describe the foundational and applied models in a setting where there is platform competition.

a. The Foundational Monopoly Platform Model

181. The Rochet-Tirole model was developed in a situation in which the operator of the two-sided platform has a monopoly and sets per-unit prices on both sides of the market to sellers and buyers.

182. In the instant case, Google controls the substantial majority of all App downloads on Android-compatible mobile devices and can thus appropriately be thought of as a platform monopolist. As a platform operator, Google has the ability to charge both buyers and sellers for using the Play Store.

183. Google's charge to consumers (buyers) can be thought of as Google's ability to charge for transactions, which I denote as P_B. As is typical for many two-sided markets, Google sets the consumer access price (in this case, a subsidy) near zero.[380] As for developers (sellers),

---

377. The model presented in Part V.E can also be applied if there are two relevant markets.

378. The subsidy referenced here is paid by Google to the consumer. The consumer still pays a positive price to the developer—albeit a lower one due to the subsidy.

379. Google limits prices to between \$0.05 and \$400.00 on the Google Play Store. Google - Play Console Help, *Supported locations for distributions to Google Play users, available at* support.google.com/googleplay/android-developer/answer/10532353?visit_id=637777015722462270-3131223409&rd=1.

380. *See, e.g.,* John M. Newman, *Antitrust in Zero-Priced Markets: Foundations*, 164 PENN LAW REVIEW 149-206 (2015). The consumer also pays for access by sharing her data with the platform operator.

Google charges a take rate, or percentage of sales, of up to 30 percent. The foundational model uses a per-unit transaction price on the seller side, which I denote as P_S, instead of a percentage take rate. In the foundational model, P_B and P_S should be understood as prices charged to consumers (buyers) and developers (sellers), respectively, for a transaction made on the platform. These prices are distinct from the price of the actual product being purchased (App downloads). Maximizing profit (by taking the derivative of the profit equation with respect to both prices) leads to an optimal pricing rule under a two-sided monopoly platform:

(V.1)
$$\frac{P_B + P_s - C}{P_B + P_s} = \frac{1}{\varepsilon_B + \varepsilon_S}$$

where $\varepsilon_B, \varepsilon_S$ are the price elasticities of demand for the buyer and seller, respectively.[381] The left-hand side of this expression represents the platform operator's per-unit margin. In maximizing its profit, the monopolist will choose to set platform prices to buyers and sellers according to their price elasticities of demand, and C represents the platform operator's incremental cost of executing a transaction. As observed in Rochet and Tirole 2003, when expressing the total price charged by the platform ($P = P_B + P_S$) and the combined elasticity with respect to both sides of the market faced by the platform ($\varepsilon = \varepsilon_B + \varepsilon_S$), Equation (V.1) simplifies to what is known as the Lerner index, the standard inverse elasticity formula:

(V.2)
$$\frac{P - C}{P} = \frac{1}{\varepsilon}$$

This expression is widely recognized in economic theory and suggests that firms with pricing power increase prices until the markup of price over marginal cost is equal to the inverse of the firm's own-price elasticity.[382]

b. Application of the Two-Sided Monopoly Platform Model to the Play Store

184. I now adapt the Rochet-Tirole model set out above to fit the current circumstances. I accommodate three key features that distinguish the adapted model from the foundational monopoly model described above. *First*, Google charges a take rate to developers on the Play Store as a percentage of developers' revenues rather than a per-unit price. *Second*, as I demonstrate later in Part V.D using standard economic principles, the take rate imposed on developers is passed through at least in part to consumers. This pass-through results in product prices that will be affected by the take rate. *Third*, through its Play Points loyalty program and other promotions, Google offers a subsidy (a negative platform price) on the consumer side.

185. Define the take rate t as the commission charged by Google to developers for using the platform (typically 30 percent with the exceptions discussed above). The per-unit amount paid to Google by the developer is equal to the take rate multiplied by the product price, which I will denote as S. For example, if an App is priced at $S = \$10$ and the take rate is 30 percent, the developer will pay 0.3*$10 = $3 to Google. Indeed, this arrangement is analogous to setting

381. Rochet & Tirole at 996-997. In mathematical terms, the elasticity of demand is defined as the percentage increase in demand divided by the percentage decrease in prices.

382. *See, e.g.,* Landes & Posner at 937 (1981).

platform prices $P_S = tS$. It is important to note that the *product* price S is also affected by the take rate, because the take rate represents a cost to developers, a significant portion of which is typically passed on to consumers in the form of higher product prices. I estimate the rate of pass-through in Section V.D and denote it here with the symbol γ. The pass-through rate is equal to the portion of an increase in costs incurred by developers (including those from increased commissions), which is passed through to consumers in the form of higher product prices. For example, if costs to a developer increase by one dollar, a pass-through rate of 0.90 means that product prices for consumers increase by $0.90. Allowing for this relationship, Equation (V.1) becomes:

(V.3)
$$\frac{P_B + tS - C}{tS + t^2 S'} = \frac{1}{\varepsilon_{B,t} + \varepsilon_{S,t}}$$

where $\varepsilon_{B,t}$ and $\varepsilon_{S,t}$ are price elasticities of demand for transactions from buyers (consumers) and sellers (developers), respectively, now taken with respect to the take rate t, which takes the place of the platform price, and $t^2 S'$ is an additional term which accounts for the effects of the take rate on the product price.[383] Appendix 3 contains a derivation of Equation (V.3).

2. The Platform Model in a Competitive Setting

a. The Foundational Competitive Model

186. When competition to the platform monopolist is introduced, both buyers and sellers can connect to more than one platform, which, as discussed above, is known as multi-homing.[384] With multi-homing, the monopolist loses some pricing power, resulting in a lower equilibrium take rate. The competitive pressure on the take rate occurs through two channels: (1) the platform's incentive to attract sellers, and (2) sellers' ability to steer buyers by way of lower product (in this case App) prices.[385]

187. All else equal, sellers will prefer to use the platform that charges a lower seller-side platform price (P_S), assuming that the alternative platform has a roughly comparable menu of products and therefore attracts a significant base of consumers. A competing platform under this assumption can therefore attract sellers away from a rival by offering a lower platform price. This first effect on platform prices, namely downward pressure in the face of competition, is analogous to the familiar forms of price competition that occur in countless industries.

188. A second effect stems from sellers' incentive to avoid a higher take rate, all things equal, while having access to the most buyers possible. Because sellers here set their own product

383. S' represents the amount by which the product price S changes when there is a change in the take rate. Appendix 3 contains further details regarding this term.

384. Rochet & Tirole at 991-992 ("In a number of markets, a fraction of end users on one or the two sides connect to several platforms. Using the Internet terminology, we will say that they 'multihome.' For example, many merchants accept both American Express and Visa; furthermore, some consumers have both Amex and Visa cards in their pockets. Many consumers have the Internet Explorer and the Netscape browsers installed on their PC, and a number of Web sites are configured optimally for both browsers.").

385. *Id.* at 1001 ("This increases demand for Platform 1 in two ways: The platform attracts new merchants…and 'steers' former multihoming merchants.").

prices, they can "steer" buyers to a platform by offering lower product prices on that platform. Steering is facilitated when a rival platform charges a lower platform price, because a seller using the platform with a lower platform price has a price differential available to lower prices and steer customers.

189. In a competitive platform setting, the platform's optimal pricing rule from Equation (V.1) becomes:[386]

(V.4) $$\frac{P_B + P_S - C}{P_B + P_S} = \frac{1}{\varepsilon_{OB} + \varepsilon_{OS}}$$

This formula now reflects the buyers' "own-brand" elasticity, ε_{OB}, and the sellers' "own-brand" elasticity, ε_{OS}. Own-brand elasticity is the change in demand for a given platform due to an increase in the price of transacting on that particular platform. This elasticity varies from the elasticity in the monopoly setting due to the presence of competition from rival platforms. In a monopoly setting, a consumer may choose not to transact in the face of a price increase but will not have the option of transacting on an alternative platform. In a competitive setting, a consumer may choose not to transact at all or may choose to transact on a competing platform. The presence of a competitive option suggests a greater elasticity of demand relative to that of the monopoly setting.

190. The own-brand elasticities cause the denominator on the right-hand side of Equation (V.4) to increase relative to the denominator in Equation (V.1). This higher denominator leads to a lower margin on the left-hand side, which implies lower equilibrium platform prices ($P_B + P_S$) in the presence of competition.

b. Application Of The Two-Sided Competitive Platform Model to The Instant Case

191. Applying the competitive model to this case results in an expression analogous to Equation (V.4):

(V.5) $$\frac{P_B + tS - C}{tS + t^2 S'} = \frac{1}{\varepsilon_{OB,t} + \varepsilon_{OS,t}}$$

As in Equation (V.3), the platform prices P_s on the left side of the expression has been replaced with its take rate analogue (tS), and there is an additional term in the denominator which accounts for the effect of a new take rate on product prices. The platform price elasticities on the right-hand side have also been replaced with their take rate analogues, now reflecting the introduction of competition ($\varepsilon_{OB,t}$ is own-brand elasticity of demand taken with respect to the take rate on the

386. *Id.* at 1004. I derive this expression by replacing market demand faced by the platform operator (in the monopoly setting) with residual demand, where residual demand is defined as market demand minus demand that is met by the platform's rivals. Rochet and Tirole model competition in the form of a duopoly and express the seller side own-brand elasticity as $\varepsilon_{OS} = \varepsilon_S/\sigma$, where σ is a single-homing index. I use the more general notation to show that in my extension of the model, I am agnostic to the number of competing platforms faced by Google, as long as there is at least one rival. Appendix 3 provides details regarding this derivation.

buyer side, and $\varepsilon_{OS,t}$ is own-brand elasticity of demand taken with respect to the take rate on the seller side). As in the foundational model, the competitive elasticity terms imply a lower take rate in this equation. Table 2 summarizes these equations, comparing the foundational framework with the extension that allows for a percentage take rate. Details of how these expressions are derived are in Appendix 3.

TABLE 2: EQUILIBRIUM EXPRESSIONS OF THE ROCHET-TIROLE MODEL APPLIED TO THE INSTANT CASE

Scenario	Foundational Model	Applied Model
Monopoly	$\frac{P_B + P_s - C}{P_B + P_s} = \frac{1}{\varepsilon_B + \varepsilon_S}$ (Eqn. (V.1))	$\frac{P_B + tS - C}{tS + t^2 S'} = \frac{1}{\varepsilon_{B,t} + \varepsilon_{S,t}}$ (Eqn. (V.3))
Competitive	$\frac{P_B + P_S - C}{P_B + P_S} = \frac{1}{\varepsilon_{OB} + \varepsilon_{OS}}$ (Eqn. (V.4))	$\frac{P_B + tS - C}{tS + t^2 S'} = \frac{1}{\varepsilon_{OB,t} + \varepsilon_{OS,t}}$ (Eqn. (V.5))

3. Calibrating the Model and Required Inputs

192. Once the model is "calibrated" in the sense that it relates the observed variables in the monopoly setting in Table 2 and solves for the unobserved variables, the model can be used to project Google's take rate in a competitive setting. I demonstrate impact by proceeding in two steps. *First*, I calibrate the Applied Model in the monopoly scenario by estimating inputs in the observed setting in which Google wields monopoly power in the Android App Distribution Market, thus satisfying Equation (V.3). The model's inputs are informed entirely by paid Apps in the Android App Distribution Market, as those are the only Apps that are priced and thus exhibit an observable own-price elasticity of demand. *Second*, I use the competitive inputs—namely, the take rate elasticities of demand—to determine a competitive take rate in a competitive (but-for) world, thus satisfying Equation (V.5). Data obtained from Google and other sources can be used in the applied monopoly and competitive models. In the descriptions below, I use the superscript ^{M} to denote inputs to the monopoly model (Equation (V.3)) and the superscript ^{C} to denote inputs to the competitive model (Equation (V.5)). My sources and methods for obtaining the monopoly scenario inputs shown in Equation (V.3) are:

- P_B^M is equal to the price "charged" by Google to consumers for transactions made on its platform in the monopoly scenario. Through its Play Point loyalty program and other promotions, Google effectively charges a small negative price to consumers. As it does in the actual world, Google would maximize its profits with respect to all Apps collectively, not App-by-App. Therefore, I use Google's average subsidy across all Apps, not individual subsidy amounts, to calculate P_B^M. I compute the value of this subsidy as the sum of all promotions paid by Google for paid Apps downloaded in the Android App Distribution Market divided by the total quantity of paid Apps downloaded in the Android App Distribution Market, per Google's transaction records.

- t^M is equal to the observed take rate, computed as the sum of all revenue retained by Google in the Android App Distribution Market divided by the sum of total revenue spent by consumers in the Android App Distribution Market. t^M therefore represents the portion of consumer spending that Google "takes" from the developer. I calculate t^M prior to extracting Google's promotional payments to consumers (promotional payments are captured by P_B^M).

- S^M is equal to the average price charged for Apps in the Android App Distribution Market (for paid App downloads only) in the monopoly setting.[387] In the monopoly model, S^M is total consumer expenditure (prior to receiving promotions from Google) in the Android App Distribution Market divided by the total quantity of paid Apps downloaded, as observed in Google's transaction records. As it does in the actual world, Google would maximize its profits with respect to all Apps collectively, not App-by-App. Therefore, I use Google's average App price across all Apps, not individual App prices, to calculate S^M.

- Marginal cost C represents the incremental cost incurred by Google in executing a transaction in the Android App Distribution Market or In-App Aftermarket. I refer to Google's financial data to infer this value, which suggests that transaction fees and direct costs that Google records for the Play Store (excluding content costs) are approximately 10.29 percent of consumer expenditures.[388]

- γ is equal to the change in the App price S charged to consumers with respect to a change in developers' costs (including the cost imposed on developers through Google's take rate), also known as the pass-through rate. This parameter is discussed in detail in Part V.D, where I estimate its value at approximately 90 percent (89.9 percent). This value implies that an increase in the take rate that adds $1.00 in extra cost to a developer will cause an increase in the price of the app product of $0.90. Mathematically, the pass-through rate is:

(V.6)
$$\gamma = \frac{\text{change in revenue}}{\text{change in costs}}$$

- S'^M represents the change in the product price resulting from a small change in the take rate. I solve for S'^M in terms of the take rate and pass-through rate: $S'^M = \frac{\gamma}{(1-t^M\gamma)} S^M$. Appendix 3 contains a derivation of this expression.

---

387. Apps that are free to download have a zero price and are therefore excluded from the analysis.

388. As I discuss in Section V. C, I estimate that Google's direct costs of sales and direct operating expenses for the Play Store (excluding irrelevant content costs for movies, television, and books) to be [redacted] percent of consumer expenditures on the Play Store for the period 2016 – 2020. In addition to transaction fees, the Play Store's direct costs of sales includes content costs, customer support, and other costs. I include all of these except content costs; these are costs Google incurs for sales of books, movies, and television, and are not part of the relevant markets here. I also include all direct operating expenses, which include payroll & stock-based compensation, as well as the following non-payroll costs: professional services, advertising and promotional expenses, equipment, and other expenses (travel and entertainment, office and related expenses). *See* GOOG-PLAY-000416245.

- $\varepsilon_{B,t}^{M}$ and $\varepsilon_{S,t}^{M}$ are the take-rate elasticities of demand for transactions in the Android App Distribution Market from consumers and developers, respectively, in the presence of Google's monopoly. $\varepsilon_{B,t}^{M}$ reflects the change in the quantity demanded by consumers for Android App Distribution Market transactions associated with a change in the take rate in a monopoly setting. A change in the take rate affects the price at which App products (paid App downloads and purchases of In-App Content) are set via pass-through, which in turn affects consumer demand. $\varepsilon_{S,t}^{M}$ reflects the change in the number of paid Apps sold by developers in response to a change in the take rate in a monopoly setting. Given the other inputs to the monopoly model, the value of $\varepsilon_{B,t}^{M} + \varepsilon_{S,t}^{M}$ is implied by Equation (V.3). Further description of these inputs is included in Appendix 3.

I hold inputs C and γ fixed between the monopoly and competitive scenarios. My sources and methods for obtaining the remaining inputs to the competitive scenario expression shown in Equation (V.5) are:

- P_B^C is equal to the price "charged" by Google to consumers for transactions made on its platform in the competitive scenario. I hold the buyer-side platform price fixed in proportion to the product price: $P_B^C = \left(\frac{P_B^M}{S^M}\right) * S^C$.[389]

- t^C is equal to the but-for (competitive) take rate. I calculate the but-for take rate by finding the value that satisfies Equation (V.5) given the remaining inputs.[390]

- S^C is the price of paid App downloads that developers would charge in a competitive scenario. S^C can be inferred if the pass-through rate is known by using Equation (V.6). In particular, plugging in the change in revenue and change in costs associated with the monopoly versus a competitive scenario:

(V.7)
$$\gamma = \frac{\text{change in revenue}}{\text{change in costs}} = \frac{(S^M - S^C) * \text{quantity}}{(t^M S^M - t^C S^C) * \text{quantity}}$$

This expression can be further simplified and re-arranged to express the competitive price S^C in terms of other inputs:

(V.8)
$$S^C = S^M \frac{1 - \gamma t^M}{1 - \gamma t^C}$$

- S'^C represents the change in the product price resulting from a small change in the take rate in the competitive setting. I solve for S'^C in terms of the take rate and pass-through rate: $S'^C = \frac{\gamma}{(1 - t^C\gamma)} S^C$. Appendix 3 contains a derivation of this expression.

389. In Section V.E, I model a scenario in which the locus of competition occurs on the buyer-side platform price P_B, resulting in a but-for buyer-side platform price that differs from the observed, monopolistic price.

390. If all the inputs to Equation (V.5) are known except for the take rate, I can solve for the take rate that satisfies the equation.

- $\varepsilon^{C}_{OB,t}$ and $\varepsilon^{C}_{OS,t}$ are the "own-brand" take-rate elasticities of demand for transactions in the Android App Distribution Market for consumers and developers, respectively, in the presence of competition. $\varepsilon^{C}_{OB,t}$ reflects the change in the quantity demanded by consumers for Android App Distribution Market transactions—from Google in particular, hence, "own-brand"—associated with a change in Google's take rate. Relative to its monopoly analogue, this parameter reflects a scenario where Google faces competition from rival platforms; as such, the parameter will be greater in magnitude than the monopoly elasticity, because the presence of a competitor allows easier defection by consumers in the presence of a price increase from Google, and thus more sensitivity. $\varepsilon^{C}_{OS,t}$ reflects the change in the quantity of transactions demanded by developers—on the Play Store in particular, hence "own-brand"—in response to a change in the take rate, again in the presence of platform (app store) competition. To inform the but-for competitive elasticities as shown in the denominator of Equation (V.5), $\varepsilon_{OB,t} + \varepsilon_{OS,t}$, I draw from the economics literature, empirical evidence of industries that have shifted from monopoly to competition. I conservatively estimate that Google's take rate elasticities shift from a value of 2.12 (in the monopoly setting, as calculated using Equation (V.3)) to 2.49 in the competitive setting. I arrive at 2.49 using the relationship between own-brand elasticity and market demand elasticity, and under the conservative assumption that Google maintains a 60 percent share of the Android App Distribution Market with an inelastic supply response from Google's rivals.[391] These inputs are defined mathematically in Appendix 3.

4. Competitive Take Rate Results

193. Table 3 summarizes the results of calculating inputs as described above. I estimate that in the but-for world, platform competition results in a competitive take rate of [redacted] percent, down from its observed value of [redacted] percent in the actual world.[392] This result is calculated from Equation (V.5), by finding the value for t that satisfies the equation, given all other inputs. As Table 3 shows, at a pass-through rate of $\gamma =$ [redacted] percent, the resulting but-for average price of paid App downloads in the Android App Distribution Market is [redacted] down from the observed price of [redacted] (net of Google's promotional expenditures to consumers). This difference results in an average overcharge to consumers of [redacted] per paid App download = [redacted]

---

391. Similar to Part V.C, *infra*, I use the relation $E_g = \frac{E_M}{S_g} + \frac{E_S(1-S_g)}{S_g}$ where E_g is Google's own-brand elasticity (reflecting price responses of both buyers and sellers), E_M is market elasticity, S_g is Google's market share, and E_S is the elasticity of supply of Google's rivals (conservatively set to zero). *See* Landes & Posner at 939-940. I conservatively assume Google maintains a 60 percent market share in a competitive market and that $E_S = 0$. AT&T saw its market share decline to approximately 60 percent by the early 1990s after losing its monopoly. *See, e.g.,* Simran Kahai, David Kaserman & John Mayo, *Is the "Dominant Firm" Dominant? An Empirical Analysis of AT&T's Market Power*, 39 JOURNAL OF LAW & ECONOMICS 499-517 (1996). This implies that buyer price elasticity of demand changes from 5.118 in the monopoly setting (estimated using Equation V.11) to 8.53 = 5.119/0.6 in the competitive setting, which translates to a competitive take rate elasticity of 2.276 (see Appendix 3 and Equation (A.19) for details on the relation between the buyer price elasticity of demand and buyer take rate elasticity of demand). I use Equation V.3 to calculate the seller (developer) take rate elasticity of demand at 0.213 in the monopoly setting. I conservatively hold the seller (developer) take rate elasticity of demand fixed at its monopoly level in the competitive (but-for) scenario The sum total of both competitive elasticities is then equal to 2.276 + 0.213 = 2.49.

392. A large portion of products sold in the Android App Distribution market were priced at either $0.99 or $1.99. When taking a 30 percent commission, Google's portion was rounded to the nearest penny (for example, taking $0.30 for a $0.99 purchase and $0.60 for a $1.99 purchase). This leads to an overall average take rate that is slightly greater than 30 percent (30.1 percent).

demonstrating impact, and it results in aggregate damages of $18.76 million (equal to $0.3047 times 61.58 million paid App downloads sold)[393] as a result of Google's restrictions in the Android App Distribution Market, across the Class Period (August 16, 2016, through December 31, 2020). As explained below, there are additional damages and impact in the In-App Aftermarket.

194. Developer-specific take rates can be computed by applying the proportion of discounts granted in the actual world to the competitive but-for take rate. For example, suppose that the overall take rate is 30 percent in the actual world. Suppose a developer has an actual take rate of 29 percent (one percentage point below the overall rate). Suppose that the overall but-for take rate is 23 percent. In this example, the developer's but-for take rate would be calculated as [23 percent] x [29 percent]/[30 percent] = 22.2 percent. The pass-through rate γ (which I set equal to 89.9 percent for this analysis) may also vary across categories of Apps. Differential pass-through rates can be readily estimated using common methods and evidence (see Part V.D.3) and inserted into the model to determine competitive but-for take rates that vary across App category, as illustrated in Part VII below. Class members who made purchases in those App categories were accordingly subject to overcharges; lower take rates associated with consumer purchases in the but-for world would be passed through in the form of lower App prices relative to the actual world.

393. Equivalently, aggregate damages can be calculated as the percentage decrease in price (equal to [redacted] multiplied by actual consumer expenditure of [redacted] (after Google's promotional expenditures), yielding aggregate damages of [redacted]

TABLE 3: ANDROID APP DISTRIBUTION MARKET IMPACT AND DAMAGES
(8/16/2016 – 12/31/2020)

Actual World (Monopoly, Eqn. (3))

#	Input	Description	Value	Source/Notes
[1]		Consumer Expenditure (US)	[redacted]	GOOG-PLAY-005535886 (US Consumers)
[2]		Google Revenue (US)		GOOG-PLAY-005535886 (US Consumers)
[3]		Google Promotional Expenditures (US)		GOOG-PLAY-005535886 (US Consumers)
[4]		Android App Distribution Market (Paid) Transactions (US)		GOOG-PLAY-005535886 (US Consumers)
[5]=[1]/[4]	$\boldsymbol{s^M}$	App Product Price		Calculated
[6]=[2]/[1]	$\boldsymbol{t^M}$	Take Rate		Calculated
[7]=-[3]/[4]	$\boldsymbol{P_B}$	Buyer-side Platform Price		Calculated
[8]=[5]+[7]	$\boldsymbol{S^M + P_B}$	App Product Price Net of Promotions		Calculated
[9]	$\boldsymbol{c}$	Marginal Cost		GOOG-PLAY-000416245
[10]	$\boldsymbol{\gamma}$	Pass-through Rate		Estimated (See Table 8)
[11]	$\boldsymbol{\varepsilon^M_{B,t} + \varepsilon^M_{S,t}}$	Take Rate Elasticities of Demand		Calculated (Eqn. (V.3))

But-For World (Competitive, Eqn. (5))

#	Input	Description	Value	Source/Notes
[12]	$\boldsymbol{s^C}$	App Product Price		Calculated (Eqn. (V.8))
[13]	$\boldsymbol{t^C}$	Take Rate		Calculated (Eqn. (V.5))
[14]=([7]/[5])* [12]	$\boldsymbol{P_B}$	Buyer-side Platform Price		Calculated
[15]=[12]+[14]	$\boldsymbol{S^C + P_B}$	App Product Price Net of Promotions		Calculated
[16]=[9]	$\boldsymbol{c}$	Marginal Cost		GOOG-PLAY-000416245
[17]=[10]	$\boldsymbol{\gamma}$	Pass-through Rate		Estimated (See Table 8)
[18]	$\boldsymbol{\varepsilon^C_{OB,t} + \varepsilon^C_{OS,t}}$	Take Rate Elasticities of Demand		Economic theory/empirical studies
[19]=[8]-[15]		Consumer Savings Per Transaction		Calculated
[20]=[19]*[4]		**Aggregate Damages**		Calculated

5. Analysis Of Similar Platforms Corroborates My Competitive Take Rates For Initial App Downloads

195. The framework described above demonstrates the economics of two-sided platforms and allows estimation of a take rate in a competitive but-for world. This model is particularly useful in the present context where the Challenged Conduct has been inherent to Google's business practices since approximately the inception of the Play Store, preventing a "before, during, and after" comparison. A comparative analysis, presented here, can be used to corroborate the results from the two-sided market model. In this section, I review take rates found in similarly situated, two-sided digital platforms. I focus on take rates from platforms where there are no (or fewer) anticompetitive restraints similar to those imposed by Google in the instant case, and the fundamentals of platform economics (connecting two sides of a market) are present. From these examples, several conclusions emerge:

- Platforms in competitive environments compete by lowering their take rates;
- Customer mobility, which hinges on the presence of substitutes and the absence of switching costs, puts downward pressure on the take rate via steering; and
- Take rates in competitive environments reflect the diminishing value offered by the platform over time following the initial matching of buyer and seller.

a. The ONE Store

196. South Korean wireless carrier SK Telecom Co. spearheaded the launch of the ONE Store, a competing mobile app store in 2016.[394] The scale of this effort to compete with Google is a testament to the barriers to entry: it involved cooperation among the three largest Korean wireless carriers (SK Telecom, KT, and LG Uplus), as well as Naver, Korea's largest search engine.[395] These parties were able to achieve near-universal availability in South Korea of the rival app store by having it pre-installed on every Android handset provided by these three companies.[396] The ONE Store now has a 14.9 percent share of payment volume among app stores in South Korea.[397] The ONE Store has managed to gain share of payment volume in large part thanks to its significantly lower take rates, as well as an aggressive points system for consumers. The ONE Store has a 20 percent take rate for developers, which is lowered to five percent if the developer uses its own payment platform.[398] ONE Store's CEO credits its lower take rates compared to Google's 30 percent rate with increasing ONE Store's presence in its domestic market and increasing the number of users purchasing app products (both paid App downloads and purchases

394. Lim Young-sin & Choi Mira, *Korea's home-grown integrated app market One Store on global outreach,* PULSE (Nov. 13, 2019), *available at* pulsenews.co.kr/view.php?year=2019&no=938924.

395. *Id.*

396. GOOG-PLAY-000005203 at GOOG-PLAY-000005264 ("Pre-installed on (virtually) every phone sold in SK").

397. Kim Eun-jung, *Korean app market ONE store eyes global alliance to compete with Google* (Dec. 1, 2019), *available at* en.yna.co.kr/view/AEN20191128004700320.

398. *Id.* ("ONE store cut the rate to 20 percent in July 2018. For app providers with their own payment platform, the firm only charged 5 percent for its service.").

of In-App Content).[399] In October 2020, ONE Store announced a 50 percent discount on commissions for small developers (those with revenue less than 5 million won per month).[400]

197. The ONE Store has been identified by Google as a competitive risk due to its lower take rate.[401] The ONE store originally charged a 30 percent commission from its launch in March 2016, and cut its take rate to the 20 percent level (five percent if developers provide their own payment platform) in July 2018 to compete against Google.[402] Developers can now (setting aside any restrictions by Google) steer their customers to the lower-cost platform via discounting prices to consumers for Apps. This episode demonstrates that multi-homing competition among app store platforms engenders competition along the take-rate dimension.

198. The scale of the alliance of the three largest wireless carriers in South Korea enabled the ONE Store to overcome the prohibitive restrictions to competition imposed by Google. Google's revenue-sharing agreements with carriers were designed to prevent such a launch of a competing app store, particularly in the United States.[403] An internal Google presentation notes that this form of competition, involving a coalition of carriers, is "[u]nlikely in the US, given market share distribution and competition amongst carriers."[404]

b. Aptoide

199. Aptoide, another app store operating worldwide, assesses a maximum take rate of 25 percent[405] and in some cases charges a take rate as low as ten percent.[406] These take rates encourage developers to steer their customers to Aptoide's lower-cost platform. This strategy has

---

399. *Id.* ("The rate cut not only helped the firm [the ONE Store] to expand its presence in the domestic market but also improved profitability with an increased number of paid users, he said. . . With the additional firepower, Lee said ONE store will bolster efforts to create an alternative global app store capable of competing with Google and enhance the app industry ecosystem. 'A monopolistic market is not healthy for both industry players and consumers,' Lee said. 'We need more competition, not only in the domestic market but also on the global scale.'").

400. ET Telecom.com, *South Korea's app market ONE store grows amid Google's Play store policy row* (Feb. 21, 2021), *available at* telecom.economictimes.indiatimes.com/news/south-koreas-app-market-one-store-grows-amid-googles-play-store-policy-row/81135498. I have not endeavored to study the difference in App prices, if any, across the One Store and Google Play Store in South Korea. To the extent App prices have converged across the two platforms, that is consistent with pass-through and steering by developers, as one would expect a lower price for an App on the One Store to induce substitution that would put downward pressure on App prices in the Google Play Store.

401. *See, e.g.,* GOOG-PLAY-000005203 at GOOG-PLAY-000005215. *See also* GOOG-PLAY-000445443 at GOOG-PLAY-000445451.

402. Kim Eun-jung, *Korean app market ONE store eyes global alliance to compete with Google* (Dec. 1, 2019), *available at* en.yna.co.kr/view/AEN20191128004700320.

403. *See, e.g.,* GOOG-PLAY-007315383 ("We take a much needed belt and suspenders approach to our Search and Play contracts to include both OEM's and carriers. In many regions, the carriers drive the preloads for phones and tablets on Android devices."); GOOG-PLAY-001055565 ("As we discussed, we are committed to not fragmenting the market. The goal is to use your master, global market to attract developers and publish content…the 'store' or channel they would see is a subset of the market made for them, but the broad market is still available at the same level of access as before."). *See also* GOOG-PLAY-001143425 ("To belay any concerns, we are absolutely not building another market.").

404. GOOG-PLAY-002011285 at GOOG-PLAY-002011289.

405. *See* Aptoide, *For Developers*, *available at* en.aptoide.com/company/developers ("Get a minimum of 75% payout rate on in-app purchases in comparison to 70% or even 50% you get with other app distributors.").

406. *See* Revenue Share, Catappult App Distribution Console, *available at* docs.catappult.io/docs/distribution-and-revenue-share.

paid off; Aptoide presently has over 300 million users worldwide.[407] Aptoide's growth is nevertheless limited by Google's restrictions—for example, consumers cannot download Aptoide through the Google Play Store and instead must go through the cumbersome side-loading process.[408] Moreover, developers are also barred from any form of steering—that is, informing consumers using the Google Play Store that they can use Aptoide for some or all of their transactions.[409]

c. Amazon

200. [redacted] Google's documents indicate that Amazon ran significant promotions on the user side. For instance, Google's 2017 "Amazon Competitor Deep Dive" indicates that Amazon was discounting Amazon Coins up to 15 percent and discounting purchases of In-App Content for Android games up to 30 percent.[411] Amazon announced in June 2021 that it will reduce its headline take rate from 30 percent to 20 percent for small developers with less than $1 million in revenues, similar to other platforms.[412] Further, Amazon will offer promotional Amazon Web Services credits that could effectively cut its take rate to 10 percent.[413]

d. PC Game Platforms

201. Despite not being a participant in the Android App Distribution Market, video game distribution platforms on PCs are similar to mobile App distribution platforms in that they also connect developers of software applications to consumers without requiring a particular console

---

407. *See* Aptoide, *About Us, available at* en.aptoide.com/company/about-us ("Aptoide is the game-changing Android App Store. With over 300 million users, 7 billion downloads and 1 million apps, Aptoide provides an alternative way to discover apps and games, with no geo-restrictions and one of the best malware detection systems in the market.").

408. Aptoide, *How to download and install Aptoide?, available at* en.aptoide.com/company/faq/how-to-download-install-aptoide.

409. *See* Google – Play Console Help, *Understanding Google Play's Payments policy*, *available at* support.google.com/googleplay/android-developer/answer/10281818?hl=en#zippy=%2Ccan-i-distribute-my-app-via-other-android-app-stores-or-through-my-website%2Ccan-i-communicate-with-my-users-about-alternative-ways-to-pay%2Ccan-i-communicate-with-my-users-about-promotions-on-other-platforms ("Within an app, developers may not lead users to a payment method other than Google Play's billing system unless permitted by the Payments policy. This includes directly linking to a webpage that could lead to an alternate payment method or using language that encourages a user to purchase the digital item outside of the app.").

410. Derek Strickland, *Apple's 30% App Store commission is 'supracompetitive,' court declares*, TWEAKTOWN (Sept. 11, 2021), *available at* www.tweaktown.com/news/81567/apples-30-app-store-commission-is-supracompetitive-court-declares/index.html. *Epic Games, Inc. v. Apple Inc.*, Case No.: 4:20-cv-05640-YGR, Rule 52 Order After Trial On The Merits (Sept. 10, 2021), at 98. [redacted]

411 GOOG-PLAY-000879194.R at GOOG-PLAY-000879204.R.

412. Sarah Perez, *Amazon's Appstore lowers its cut of developer revenue for small businesses, adds AWS credits*, TECHCRUNCH (June 17, 2021), *available at* techcrunch.com/2021/06/17/amazons-appstore-lowers-its-cut-of-developer-revenue-for-small-businesses-adds-aws-credits/.

413. Mike Peterson, *Amazon following Apple & Google's lead, cutting app store commissions*, APPLEINSIDER (June 17, 2021), *available at* appleinsider.com/articles/21/06/17/amazon-following-apple-googles-lead-cutting-app-store-commissions.

(like an Xbox or PlayStation).[414] The three dominant platforms through which PC games are bought and sold are Steam, Epic, and Microsoft.[415] Indeed, Google has noted that a 20 percent take rate would bring "Play rev share in line with upper end of desktop gaming stores,"[416] although relatively recent developments suggest even lower take rates among PC game platforms. The Epic Games Store was launched in December 2018 with a take rate of 12 percent.[417] Microsoft announced a reduction from 30 percent to 12 percent for games sold through its store, beginning August 1, 2021.[418] Effective October 2018, Steam also announced a take-rate reduction from 30 percent to a tiered system: 30 percent for the developer's first $10 million in revenue, 25 percent for sales between $10 and $50 million, and 20 percent for sales more than $50 million.[419] Discord, a PC game platform monitored by Google, imposes a ten percent revenue share.[420]

e. PC App Stores

202. Effective August 1, 2021, Microsoft charged a 12 percent take rate for consumer non-game apps sold in the Microsoft Store (on devices other than Xbox and those using Windows 8), reduced from 15 percent.[421] Importantly, these commissions only apply when the developer is using the Microsoft commerce platform to "support the purchase of your App or any in-App Products" (analogous to Google's billing system).[422] Also as of August 1, 2021, Microsoft charged a *zero* percent take rate for non-game apps downloaded through the Windows 11 Store if the developer chose to use its own or a third-party commerce platform to facilitate in-app purchases.[423]

---

414. Take rates for video games played on consoles such as Xbox and Playstation may reflect the cost recovery of the hardware.

415. Steam is estimated to control roughly three quarters of PC gaming sales, followed by Epic (between two and 15 percent) and Microsoft. *See, e.g.,* Kyle Orland, *Humble Bundle creator brings antitrust lawsuit against Valve over Steam*, ARS TECHNICA (Apr. 30, 2021), *available at* arstechnica.com/gaming/2021/04/humble-bundle-creator-brings-antitrust-lawsuit-against-valve-over-steam/.

416. GOOG-PLAY-000542516 at GOOG-PLAY-000542529.

417. Epic Games, *The Epic Games store is now live* (Dec. 6, 2018), *available at* www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live ("The Epic Games store is now open, featuring awesome high-quality games from other developers. Our goal is to bring you great games, and to give game developers a better deal: they receive 88% of the money you spend, versus only 70% elsewhere. This helps developers succeed and make more of the games you love.").

418. Tom Warren, *Microsoft shakes up PC gaming by reducing Windows store cut to just 12 percent*, THE VERGE (Apr. 29, 2021), *available at* www.theverge.com/2021/4/29/22409285/microsoft-store-cut-windows-pc-games-12-percent.

419. Brittany Vincent, *Valve Introduces New Revenue Split Changes For Steam Sales,* VARIETY (Dec. 3, 2018), *available at* variety.com/2018/gaming/news/valve-revenue-split-changes-1203078700/.

420. GOOG-PLAY-007329076 at GOOG-PLAY-007329084.

421. Microsoft Store, *App Developer Agreement Version 8.7* (Effective July 28, 2021), *available at* query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4OG2b ("Fifteen percent (15%) of Net Receipts for any Apps (and any In-App Products in such Apps, including) that are not listed in Section 6(b)(iii) below. ii. For all Net Receipts generated on or after August 1, 2021: Twelve percent (12%) of Net Receipts for any Games (and any in-App Products in such Games) that are not listed in Section 6(b)(iii). iii. Thirty percent (30%) of Net Receipts for: 1. all Apps and In-App Products acquired by Customers in the Microsoft Store on an Xbox console and billed to such Customers on a non-subscription basis; 2. all Games (and In-App Products in Games) acquired by Customers in the Microsoft Store on an Xbox console; and 3. all Apps and In-App Products acquired by Customers in the Microsoft Store on Windows 8 devices; or Microsoft Store on Windows Phone 8 devices.").

422. *Id.* at 13-14 ("Commerce Platform Requirements. Purchases made on a third-party commerce engine are not subject to the Store Fee, but are still required to comply with our Certification Requirement.").

423. Giorgio Sardo, General Manager – Microsoft Store, *Building a new, open Microsoft Store on Windows 11*, MICROSOFT WINDOWS BLOGS (Jun. 24, 2021), *available at* blogs.windows.com/windowsexperience/2021/06/24/building-a-new-open-microsoft-store-on-windows-11/.

More specifically, the Microsoft Store charges game developers 12 percent of revenue; non-game app developers pay 15 percent of revenue if they use Microsoft platform for their in-app transactions, but zero percent if they do not:

> Many developers love the Microsoft Commerce platform because of its simplicity, global distribution, platform integration and its competitive revenue share terms at 85/15 for apps and 88/12 for games. Starting July 28, app developers will also have an option to bring their own or a third party commerce platform in their apps, and if they do so they don't need to pay Microsoft any fee. **They can keep 100% of their revenue**."[424]

203. The Microsoft PC app store faces competition from direct downloads—consumers can easily discover a new application on the Internet and download it to the personal computer without using Microsoft as an intermediary. Given the competition from direct app downloads, Microsoft only charges a take rate when the services of matchmaking—connecting the consumer to the app—and billing services are provided.

f. Other Examples

204. Additional examples of take rates more competitive than Google's abound in other, similarly situated industries with two-sided platforms. In independent online publishing, one of the leading platforms, Substack, which brings together writers and readers, takes a ten percent commission from writers, recognizing the low switching costs: "Moving one's email list away from Substack is simple, so the firm lets writers keep 90% of their revenues."[425] This ease of mobility increases writers' elasticity of supply, which puts downward pressure on the take rate. Revue, a competitor to Substack now owned by Twitter, charges only a five percent take rate.[426] Google's own Chrome web store, which provides extensions, themes, and apps associated with its browser, charges a five percent take rate, recognizing the value of attracting developers who might otherwise produce content for other browsers.[427] Take rates for online retail from vendors such as

---

424. *Id.* (emphasis added); *see also* Alex Hern, *Microsoft to let developers keep all their Windows app store revenue*, THE GUARDIAN (June 25, 2021), *available at* www.theguardian.com/technology/2021/jun/25/microsoft-let-developers-keep-all-windows-app-store-revenue#:~:text=Currently%2C%20developers%20who%20sell%20apps,the%20revenue%20with%20the%20company. ("As part of the shift to Windows 11, unveiled on Thursday, the company will allow developers to use their own payment systems on apps they sell through the Windows store. Those who do will not have to pay a penny to Microsoft."). Thus far, Microsoft has declined to unbundle its billing system for game developers: "A different set of rules apply for game developers: their share is lower, at 12%, but they will not be given the option of using their own payment processors." *Id.*

425. *The new rules of the 'creator economy'*, ECONOMIST (May 8, 2021), *available at* www.economist.com/briefing/2021/05/08/the-new-rules-of-the-creator-economy.

426. Max Willens, *Cheat sheet: Twitter's acquisition of Revue heats up the battle of the inbox*, DIGIDAY (Jan. 27, 2021), *available at* digiday.com/media/cheat-sheet-twitters-acquisition-of-revue-heats-up-the-battle-of-the-inbox/ ("Revue will remain a separate brand, but Twitter will provide the resources to make Revue more competitive with other newsletter platforms; the commission Revue takes on all consumer revenue has been reduced to 5%, half of what Substack charges. All of the Pro features for Revue will be freely available to all Revue users as well. Twitter will also help Revue hire more people across research, design and engineering.").

427. D. Melanson, *Google makes Chrome Web Store available worldwide, adds in-app purchases and flat five percent fee,* ENGADGET (May 11, 2011), *available at* www.engadget.com/2011-05-11-google-makes-chome-web-store-available-worldwide-adds-in-app-pu.html.

Amazon, eBay, and Etsy range from eight to fifteen percent with a small additional lump sum on the order of $0.30-$0.99.[428]

205. Table 4 offers a non-comprehensive summary of take rates in comparable competitive digital platform environments. Google's competitive but-for take rate from my two-sided platform model of 23.4 percent is corroborated by rates charged by competitive mobile app stores (18 to 25 percent), and is conservative compared to the take rates imposed by other platforms in more competitive industries. Finally, take rates of 15 percent [redacted] offered by Google pursuant to LRAP (and similar programs) also provide a reasonable approximation of the but-for take rate. Record evidence shows that LRAP offered a 15 percent take rate to induce premium subscription video streaming services with viable non-Google billing options (such as [redacted], and others) to adopt Google Play Billing in their apps.[429] It therefore provides a valid competitive benchmark take rate for developers with take rates at (or close to) Google's standard 30 percent take rate in the actual world (that is, the vast majority of developers).[430]

---

428. *See, e.g.,* Hung Truong, *Compare 9 Online Marketplace Fees* (Sept. 18, 2018), *available at* sellerzen.com/compare-9-online-marketplace-fees.

429. PX693, at GOOG-PLAY-000578301.R. *See, also* Defendants' Responses and Objections to Developer Plaintiffs' First Set of Interrogatories at 13 ("Google maintains several developer programs that lower the service fee earned by Google on apps and in-app purchases distributed by those developers on Google Play. Programs that U.S. developers participate in include "Living Room Accelerator Program" ("LRAP"), LRAP++, Audio Distribution Accelerator Program ("ADAP"), and Subscribe with Google ("SwG").") Google created LRAP for developers "who had video subscription apps" to encourage them to help "build up [Google's] android living room experience," including to integrate a product known as "Cast" into their mobile apps which "would then allow their content to be seen on TVs" and to integrate with Google Play billing. *See* Rosenberg Dep. at 261:11-262:4. Google offered a [redacted] percent take rate to induce premium subscription video streaming developers such as [redacted], and others to adopt Google's billing products in their apps. *See, e.g.,* GOOG-PLAY-000338849 at GOOG-PLAY-000338888; GOOG-PLAY-004714797; GOOG-PLAY-004717237; Defendants' Responses and Objections to Developer Plaintiffs' First Set of Interrogatories at 14-15; GOOG-PLAY-0006998204.R at GOOG-PLAY-0006998206.R. Further, under the LRAP++ program, Google has offered take rates of [redacted] (*see* GOOG-PLAY-000442329 at GOOG-PLAY-000442345- GOOG-PLAY-000442346; GOOG-PLAY-004717237) and [redacted] (see GOOG-PLAY-000338849 at GOOG-PLAY-000338888; GOOG-PLAY-006998204.R at GOOG-PLAY-0006998206.R).

430. As explained in n. 376, *supra*, the relatively few developers who paid reduced take rates in the actual world would also have paid reduced take rates relative to the overall but-for take rate. For example, Disney would have more options for billing services than a typical developer in the but-for world, just as it does in the actual world.

TABLE 4: BENCHMARK TAKE RATES

Category	Benchmark		Comparable Take Rate
Mobile App Stores	(1)	Aptoide	10-25%
	(2)	ONE Store	5-20%
	(3)	Amazon	18%
PC App Stores	(4)	Microsoft (non-games)	12-15%
PC Games	(5)	Steam (Valve)	20-30%
	(6)	Epic	12%
	(7)	Microsoft Store	12% effective 8/1/2021
Online Retail	(8)	Amazon	8-15% + $0.99/item or $39.99/month
	(9)	eBay	12.55% + $0.35
	(10)	Etsy	8% + $0.45
	(11)	Google	0% (previously 5-15%)
	(12)	Poshmark	20% (for over $15, $2.95 flat fee for under $15 sale)
	(13)	Walmart	6-15%
Online Publishing	(14)	Substack	10% + credit card fee
	(15)	Revue (Twitter)	5%

Sources: (1) Aptoide – Catappult, *Revenue Share*, *available at* docs.catappult.io/docs/distribution-and-revenue-share (2) Korean app market ONE store eyes global alliance to compete with Google (Dec. 1, 2019), *available at* en.yna.co.kr/view/AEN20191128004700320; GOOG-PLAY-007329076 at GOOG-PLAY-007329084 (showing a 20 percent take rate, originally at 30 percent); (3) Derek Strickland, *Apple's 30% App Store commission is 'supracompetitive,' court declares*, TWEAKTOWN (Sept. 11, 2021), *available at* www.tweaktown.com/news/81567/apples-30-app-store-commission-is-supracompetitive-court-declares/index.html. (showing Amazon's effective take rate of 18.1%); (4) Microsoft Store App Developer Agreement Version 8.7 (Effective July 28, 2021), *available at* query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4OG2b; (5) Brittany Vincent, *Valve Introduces New Revenue Split Changes For Steam Sales*, VARIETY (Dec. 3, 2018), *available at* variety.com/2018/gaming/news/valve-revenue-split-changes-1203078700/; (6) Epic Games, *The Epic Games store is now live* (Dec. 6, 2018), *available at* www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live; (7) Tom Warren, *Microsoft shakes up PC gaming by reducing Windows store cut to just 12 percent*, THE VERGE (Apr. 29, 2021), *available at* www.theverge.com/2021/4/29/22409285/microsoft-store-cut-windows-pc-games-12-percent; (8) Amazon, *Let's talk numbers*, *available at* sell.amazon.com/pricing; (9) eBay, *Understanding selling fees, available at* pages.ebay.com/seller-center/get-started/seller-fees.html; (10) Etsy, *Sell, available at* www.etsy.com/sell (Etsy charges a $0.20 listing fee. When a product is sold, they charge a 5% transaction fee, paired with a 3% + $0.25 payment processing fee); (11) Google Merchant Center Help, *New 0% commission fee for selling on Google through Shopping Actions in the US* (July 23, 2020), *available at* support.google.com/merchants/answer/9977875?hl=en; Bryan Falla, *Google Shopping Actions Commission Rates*, GODATAFEED (Oct. 22, 2019), *available at* www.godatafeed.com/blog/google-shopping-actions-commission-structure; (12) Poshmark, *What are the fees for selling on Poshmark, available at* support.poshmark.com/s/article/297755057?language=en_US; (13) Walmart Marketplace, *Referral Fees, available at* marketplace.walmart.com/referral-fees/; (14) Substack, *Going Paid, available at* substack.com/going-paid; (15) Tom McKay, *Twitter Wants to Be Substack Now*, GIZMODO (Jan. 26, 2021), *available at* gizmodo.com/twitter-wants-to-be-substack-now-1846136057.

C. Removing Google's In-App Aftermarket Restrictions Would Put Downward Pressure on the Take Rate Google Imposes on Developers for In-App Content

206. Relative to the value provided by the developer, the value that the Play Store contributes by matching a consumer with an App dissipates over time. That is because once a consumer has found an App on the Play Store, the match has been made. Any value added through

the purchase of In-App Content is added entirely by the developer. Google's own documents recognize this.[431]

207. I understand that all of the In-App Aftermarket services that developers are currently forced to use from Google (owing to Google's In-App Aftermarket restrictions) can actually be performed by a third party or the developer itself completely independently of Google. For example, there exists a well-established industry of competitive payment processors in the business of facilitating online transactions.[432]

208. In the competitive but-for world without Google's restrictions, developers could choose their own provider of services in the In-App Aftermarket. Alternatively, developers would be able to offer consumers the choice of selecting from an array of competitive options to provide In-App Aftermarket Content. Elementary economics dictates that this would place downward pressure on Google's take rate, pushing it closer to the marginal cost of providing any services associated with In-App Content. Developers having the ability to steer consumers to lower-cost competitors would reinforce this downward pressure, an outcome that Google has modeled in [redacted][433] As explained below, standard economic methods common to the Class can be used to conservatively estimate the extent to which Google's take rate for services in delivering In-App Content would fall when Google's restrictions are removed.

1. A Standard Economic Model of Competition in the In-App Aftermarket

209. To the extent that a competitive In-App Aftermarket would be characterized by homogenous commodity services (payment for and distribution of In-App Content) offered by various competitive rivals with few barriers to entry or expansion, standard economic principles prescribe that Google would be unable to charge a premium for these services.[434] If Google attempted to charge developers anything in excess of the competitive market price for In-App Aftermarket services, then developers would switch to a competitor providing identical services at lower cost, rendering Google's attempted price increase unprofitable.[435] Thus, to the extent that the competitive In-App Aftermarket is characterized by competition for a commoditized service, Google's equilibrium take rate in the In-App Aftermarket would fall to the marginal cost of serving that market. As explained below, my economic model of the In-App Aftermarket conservatively allows Google to charge a substantial markup above marginal cost, even in a more competitive world.

210. Record evidence shows that Google's 30 percent take rate in the In-App Aftermarket cannot be justified by the costs of serving that market. As early as 2009, Google recognized that "30% is an arbitrary fee > the transaction cost to GOOG (2%)" and noted that "in competitive landscape may drive developers away from platform."[436] In another document, Google

---

431. *See, e.g.,* GOOG-PLAY-003335786 at GOOG-PLAY-003335805 (describing Google's declining contribution to perceived value over time as applied to games).

432. *See, e.g.,* Jonas DeMuro & Brian Turner, *Best payment gateways of 2021*, TECH RADAR (Apr. 20, 2021), *available at* www.techradar.com/best/best-payment-gateways; *see also* Table 7 below (listing various competitive payment processors).

433. *See* Part V.C.2 below; *see also* GOOG-PLAY-006829073.R at GOOG-PLAY-006829085.R (assessing "Dev incentive to steer user choice").

434. *See, e.g.,* MANKIW at 268-284.

435. *Id.*

436. GOOG-PLAY-004630018.R at GOOG-PLAY-004630024.R.

contemplated consumers choosing a competitive payment processor, described the "Core Issue [:] 30% is too high,"[437] and showed "market rates" for payment processing, including "PSPs [Payment Service Providers] that focus on simplicity and ease of integration," such as Stripe and PayPal, which charged "30c + 2.9%."[438] The same Google document calculated Google's average cost of payment processing at just [redacted] percent of customer spend for the top 5,000 developers. Google's estimate of its own payment-processing costs are below corresponding charges for two competitive payment processors (Stripe and Adyen).[439] Another Google document reports that transaction costs came to [redacted] percent of consumer spend in the first five months of 2021.[440]

211. Financial data produced by Google allow me to estimate the Play Store's global transaction costs as a percentage of global customer spend in the In-App Aftermarket and the Android App Distribution Market (Google's financial data do not distinguish between the two markets).[441] In 2020, Google incurred worldwide aggregate transaction fees—including from credit cards, direct carrier billing, chargebacks, and gift cards—totaling [redacted][442] In that same year, Google's global revenue from commissions earned in the In-App Aftermarket and the Android App Distribution Market came to [redacted][443] Although the Play Store's financials do not include consumer expenditures, the consumer expenditures that gave rise to this revenue can be estimated at [redacted][444] Total transaction fees as a percentage of consumer spend on the Play Store is therefore [[redacted] percent. I obtain a similar result ([redacted] percent) if I apply the same calculations to the Play Store's aggregate financials over the most recently available five-year period (2016-2020).[445]

212. Even if I expand the calculation to include all the direct costs of sales that Google records for the Play Store (excluding irrelevant content costs for movies, television, and books), as well as all direct operating expenses, I calculate all of these costs came to [redacted] percent of consumer expenditures on the Play Store for the period 2016 – 2020.[446] Accordingly, Google's cost of providing In-App Aftermarket services can be conservatively estimated at [redacted] percent

437. GOOG-PLAY-006829073 at GOOG-PLAY-006829079.

438. GOOG-PLAY-006829073.R at GOOG-PLAY-006829097.R.

439. *Id.* at GOOG-PLAY-006829076.R (showing Google's average [redacted] percent of consumer spend, compared with 3.1 percent for Ayden and 6.2 percent for Stripe. These calculations exclude DCB [Direct Carrier Billing] and GC [Google Cloud]). When DCB and GC are included, Google estimates that its payment processing costs are below [redacted] for 2,300 out of the top 5,000 developers, and below [redacted] for the vast majority of developers. *Id.* at GOOG-PLAY-006829075.R.

440. GOOG-PLAY-007617587 ("Summary" tab); *see also* "FOP Cost Rates by country" tab (showing country blended rate of [redacted] percent for the United States).

441. GOOG-PLAY-000416245. Because these are global financial data, they are not comparable to the revenue statistics in Table 3 above and in Table 5 below, which are limited to the United States.

442. GOOG-PLAY-000416245.

443. *Id.* Excludes irrelevant revenue streams such as movies, books, and television.

444. According to App Revenue Metrics data produced by Google, Google's overall take rate in the In-App Aftermarket and the Android App Distribution Market came to [redacted] percent in 2020. *See* GOOG-PLAY-005535887; GOOG-PLAY-005535886.

445. GOOG-PLAY-000416245.

446. In addition to Transaction Fees, the Play Store's Direct Costs of Sales includes Content Costs, Customer Support, and Other. I include all of these except Content Costs; these are costs Google incurs for sales of books, movies, and television, and are not part of the relevant markets here. I also include all Direct Operating Expenses, which include Payroll & Stock-based Comp (SBC), as well as the following Non-Payroll costs: Prof Services, A&P, Equipment, and Other (T&E, Office & Related). *Id.*

of consumer expenditures.[447] This implies that Google's standard 30 percent take rate vastly exceeds its marginal costs ([redacted] percent of revenues), confirming that Google is exercising market power.

213. Google may argue that it would have retained some brand loyalty in the In-App Aftermarket, conferring a degree of pricing power in a competitive world, and thus a deviation from homogenous-product competition contemplated above. In that case, standard economics shows that Google's profit-maximizing price for In-App Aftermarket services would be determined by Google's firm-specific price elasticity of demand (as well as marginal costs).[448] The firm-specific demand elasticity is the percentage decrease in demand for Google's In-App Aftermarket services resulting from a one percent increase in price.[449] Google's profit-maximizing price for In-App Aftermarket services is given by the standard inverse elasticity formula, shown in the equation below.[450]

$$\left(P-C\right)/P = 1/E_g \tag{V.9}$$

where E_g represents Google's firm-specific demand elasticity for In-App Aftermarket services, P represents the price for In-App Aftermarket services, and C represents Google's marginal cost of providing In-App Aftermarket services. It bears noting that this elasticity of demand for Google's In-App Aftermarket services (Google Play Billing) is different from the elasticities of demand used in the two-sided model of the Android App Distribution Market for the Play Store.

214. As explained in Landes and Posner's seminal paper, Google's firm-specific demand elasticity is related to the market demand elasticity as follows:[451]

$$E_g = E_M / S_g + E_s\left(1 - S_g\right)/ S_g \tag{V.10}$$

Above, E_M is the market demand elasticity for In-App Aftermarket services—that is, the percentage decrease in the market-wide quantity demanded resulting from a one percent market-wide increase in price. The term E_s is the elasticity of supply of Google's rivals—that is, the percentage increase in the quantity supplied by Google's rivals, given a one percent increase in Google's price. Finally, S_g is Google's market share. For example, if Google's market share is 100 percent (S_g =1), the equation collapses to $E_g = E_M$. In that scenario, Google's firm-specific elasticity is the same as the market elasticity, because Google would be a monopolist (in the strict economic sense of being literally the only supplier). In contrast, when Google's market share falls below 100 percent, its firm-specific demand elasticity exceeds the market demand elasticity. By the standard

---

447. Google's cost of providing In-App Aftermarket services is certainly no more than fifteen percent of consumer expenditures, the rate that Google charges to all subscription Apps, effective January 1, 2022. *See* Sameer Samat - Vice President, Product Management, *Evolving our business model to address developer needs*, Android Developers Blog (Oct. 21, 2021), *available at* android-developers.googleblog.com/2021/10/evolving-business-model.html ("To help support the specific needs of developers offering subscriptions, starting on January 1, 2022, we're decreasing the service fee for all subscriptions on Google Play from 30% to 15%, starting from day one.").

448. *See, e.g.,* Landes & Posner at 939-940.

449. *Id.*

450. *Id. See also* Jerry Hausman & Greg Leonard, *Efficiencies from the Consumer Viewpoint,* 17(3) GEORGE MASON LAW REVIEW 707, 709 (1999) [hereafter Hausman & Leonard].

451. Landes & Posner at 944-945.

inverse-elasticity formula in equation V.9 above, Google's profit-maximizing price under competition is lower than the monopoly price for In-App Aftermarket services.

215. In the actual world, Google's share of the In-App Aftermarket is close to 100 percent,[452] because Google has prevented competitive entry by forcing developers to purchase from Google In-App Aftermarket services (authorization of In-App Content and payment processing), typically priced at 30 percent of developers' In-App Aftermarket revenue. In a competitive but-for world, elementary economic principles dictate that competitors would enter the market and charge a lower take rate to developers, diverting business from Google and pushing Google's price downward toward marginal cost.[453]

216. Economists have demonstrated empirically that previously monopolistic (or dominant) firms faced with competitive entry lose both market share and pricing power. For example, when AT&T lost its monopoly in long-distance telephone service pursuant to a 1982 divestiture order, it lost substantial market share, and long-distance telephone prices fell substantially, despite any brand loyalty that AT&T may have enjoyed over other long-distance entrants such as MCI.[454] In an article published in the *Journal of Law & Economics*, the authors found that AT&T, which had previously enjoyed a government-sanctioned monopoly, saw its market share decline to approximately 60 percent by the early 1990s.[455] The supply elasticity of AT&T's competitors was estimated at 4.38, consistent with evidence that barriers to entry and expansion in the long-distance market were relatively low during the post-divestiture period.[456] Applying equation V.10 above, the authors calculated that AT&T's firm-specific demand elasticity at between 3.73 and 7.81, which implied price-cost markups of between 13 and 29 percent.[457] These markups are below those found in a range of other industries throughout the economy, indicating that competition had substantially eroded AT&T's market power in the interstate long-distance market.[458] In the absence of competition, AT&T's profit-maximizing prices for long-distance service would have been substantially higher, particularly given that market demand for long-distance service is relatively insensitive to price.[459]

---

452. *See, e.g.*, Android Developers Blog, *Listening to Developer Feedback to Improve Google Play* (Sept. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html ("Less than 3% of developers with apps on Play sold digital goods over the last 12 months, and of this 3%, the vast majority (nearly 97%) already use Google Play's billing. But for those who already have an app on Google Play that requires technical work to integrate our billing system, we do not want to unduly disrupt their roadmaps and are giving a year (until September 30, 2021) to complete any needed updates. And of course we will require Google's apps that do not already use Google Play's billing system to make the necessary updates as well.").

453. *See, e.g.*, MANKIW at 270-282.

454. *See, e.g.,* Simran Kahai, David Kaserman & John Mayo, *Is the "Dominant Firm" Dominant? An Empirical Analysis of AT&T'S Market Power*, 39 JOURNAL OF LAW & ECONOMICS 499-517 (1996) [hereafter Kahai et al.]. *See also* Jeffrey Eisenach and Kevin Caves, *What Happens When Local Phone Service Is Deregulated?* REGULATION 34-41 (2012) at 35 ("There is no disagreement, however, that long distance prices have fallen sharply since liberalization. As shown in Figure 1, in real terms, the price of long-distance service fell by more than 70 percent between 1984 and 2006.").

455. Kahai et al. at 510. This reflects AT&T's output-based market share. Its asset-based market share was even lower, at approximately 40 percent. *Id.*

456. *Id*. at 508.

457. *Id*. at 510 ("The corresponding values of the Lerner index…are 0.29 and 0.13.").

458. *Id.* at 510-513.

459. *Id*. at 509 (reporting market demand elasticities between 0.49 and 0.75).

217. Similarly, an econometric analysis of the historically dominant Aluminum Company of America (Alcoa) found that Alcoa's pricing power declined significantly in the postwar period, despite substantial barriers to entry and expansion by competitive rivals.[460] The authors estimated the supply elasticity for Alcoa's rivals in the aluminum industry at just 1.4.[461] This was indicative of the substantial capital requirements for primary aluminum producers,[462] and particularly the "extraordinarily high" cost of entry at an efficient scale.[463] Nevertheless, Alcoa's residual demand elasticity was estimated at 8.3, indicating that Alcoa's pricing power, much like AT&T's, had substantially eroded.[464] The authors used the same formula given in equation V.10 above to estimate Alcoa's residual demand elasticity: The market demand elasticity for aluminum was estimated at 2.0.[465] Alcoa's capacity-based market share was approximately 35 percent during the relevant time period.[466] Applying equation V.10, this resulted in a relatively high firm-specific elasticity for Alcoa of 8.3.[467] This relatively high price sensitivity yields a correspondingly low price-cost markup of 12 percent.[468] The authors concluded that, despite the supply constraints faced by Alcoa's rivals, "the aluminum industry has entered a much more competitive market structure in the post-war period."[469] In the absence of competitive entry, Alcoa would have been able to command price-cost markups of approximately 50 percent (equal to $1/E_M = 1/2.0$) rather than 12 percent.

218. I apply this same standard economic framework to modeling the but-for take rate in the In-App Aftermarket. These calculations are summarized in Table 5. As seen below, U.S. consumer expenditures in the In-App Aftermarket came to [redacted] between mid-August 2016 (the beginning of the Class Period) and the end of 2020. Over this timeframe, Google collected [redacted] in U.S. commissions, resulting in a take rate in the actual world of [redacted] percent. Total U.S. transaction volume was [redacted] implying an average consumer price per transaction of [redacted].[470] Google received [redacted] percent of this price, or [redacted] per transaction. Google's marginal cost per transaction is conservatively estimated at [redacted] percent of the average consumer price, or [redacted] per transaction, which yields a markup of price over cost of [redacted] percent. By the equation (V.9) above, Google's own-firm elasticity in the In-App Aftermarket is [redacted]. I obtain a comparable result of [redacted] if I estimate Google's own-firm elasticity econometrically; here I conservatively use the lower of the two elasticity estimates.[471] By equation

460. Sheng-Ping Yang, *Identifying a dominant firm's market power among sellers of a homogeneous product: an application to Alcoa*, 34 APPLIED ECONOMICS 1411-1419 (2002).
461. *Id.* at 1416.
462. *Id.* at 1412.
463. *Id.* at 1418.
464. *Id.* at 1417.
465. *Id.* at 1416.
466. *Id.* at 1417.
467. Equal to 2.0/0.35 + 1.4*(1 - 0.35)/0.35.
468. Equal to 1/8.3 = 0.12.
469. *Id.* at 1418.
470. Average revenue is mathematically equivalent to price per unit. *See, e.g.,* MANKIW at 270 ("Average revenue is total revenue (*P* x *Q*) divided by the quantity (*Q*). *Therefore, for all types of firms, average revenue equals the price of the good.*") (emphasis in original).
471. In Appendix 5, I empirically estimate Google's actual-world own-firm demand elasticity for the In-App Aftermarket at [redacted]. My results are not highly sensitive to using this estimate, instead of the own-price elasticity of [redacted] calculated here. Aggregate damages would increase by approximately [redacted] percent if I were to use [redacted] instead of [redacted] as the own-firm demand elasticity. Here I conservatively employ the [redacted] estimate.

(V.10) above, the market demand elasticity in the In-App Aftermarket is [redacted][472] The values of these inputs in the actual world are summarized in the first panel of Table 5 below.

TABLE 5: IN-APP AFTERMARKET IMPACT & DAMAGES (US, 8/16/2016 – 12/31/2020)

#	Input	Value	Source/Notes
Actual World			
[1]	Consumer Expenditure (US; Net of Discounts)	[redacted]	GOOG-PLAY-005535886 (US Consumers; net of discounts)
[2]	Google Revenue (US)		*Id.*
[3] = [2]/[1]	Take Rate		Calculated
[4]	Quantity (Transactions)		GOOG-PLAY-005535886 (US Consumers)
[5] = [1]/[4]	Consumer Price		Calculated
[6] = [5]*[3]	Google Price		Calculated
[7] = 0.1029*[5]	Google Marginal Cost		GOOG-PLAY-000416245 (equal to 10.29 percent of consumer expenditure. Includes all Direct COS & Direct OpEx)
[8] = ([6] - [7])/[6]	Google Margin		Calculated
[9] = 1/[8]	Google Own-Firm Demand Elasticity		Calculated
[10]	Google Market Share		*See, e.g.*, android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html
[11] = [10]*[9]	Market Demand Elasticity		Calculated
Absent Google's Restrictions			
[12]	Google Market Share		Economic principles/empirical studies
[13]	Competitor Supply Elasticity		Economic principles/empirical studies
[14] = [11]/[12] + [13]*[1 - 12]/[12]	Google Own-Firm Demand Elasticity		Calculated
[15] = 1/[14]	Google Margin		Calculated
[16] = [7]/[1 - [15]]	Google Price Per Transaction		Calculated
[17] = [6] - [16]	Total Savings Per Transaction		Calculated
[18]	Pass-Through Rate		Economic principles/econometric estimates. *See* Part IV.C, *infra*.
[19] = [18]*[17]	Consumer Savings Per Transaction		Calculated
[20] = [5] - [19]	Consumer Price Per Transaction		Calculated
[21] = [16]/[20]	Take Rate		Calculated
[22] = [4]*[19]	**Aggregate Damages**		Calculated

472. In the actual world, $E_s = 0$ because competitive rivals are constrained by Google's restrictions. Therefore, $E_M = E_g S_g$. *See, e.g.*, MICHAEL KATZ AND HARVEY ROSEN, MICROECONOMICS 3rd ed. 329-330 (Irwin/McGraw-Hill 1998).

219. The values for the parameters in the competitive but-for world are summarized in the second panel of Table 5 above. Even in the presence of substantial competition, I assume conservatively that Google would have retained a substantial market share of 60 percent in the In-App Aftermarket. As noted above, this was approximately AT&T's market share in the long-distance market after competitive entry.[473] It is also substantially above Alcoa's market share after competitive entry by capacity-constrained rival aluminum manufacturers (approximately 35 percent).[474] This estimate is also conservative in relation to market share and concentration statistics for e-commerce markets, in which the payment method is generally not tied to the rest of the transaction: There exists a range of payment methods accepted in U.S. e-commerce markets, from credit and debit cards (Visa, Mastercard, etc.) to digital wallet services (such as Amazon Payments, PayPal, Square, and others).[475] Credit and debit cards account for approximately 58 percent of e-commerce transactions; the second largest payment method is digital wallets, at 25 percent.[476] Visa, the largest credit and debit platform, has a market share of 60 percent.[477] Visa's share of e-commerce payments can therefore be estimated at approximately [58 percent] x [60 percent] = 35 percent. Within the second largest category (digital wallet services), the largest firm is PayPal, with a market share of approximately 55 percent.[478] PayPal's share of e-commerce payments can therefore be estimated at approximately [55 percent] x [25 percent] = 13.75 percent.[479] Thus, my analysis assumes that, in a more competitive world, Google would command a substantially greater market share than Visa or PayPal in e-commerce.

220. In the instant case, the elasticity of supply of Google's would-be rivals in the market for In-App Aftermarket services cannot be measured directly, because Google has foreclosed entry and expansion by rivals. In Table 5 above, I set $E_s = 4.38$, based on the supply elasticity for AT&T's long-distance competitors estimated econometrically in the literature.[480] Using equation V.10, Google's competitive own-firm demand elasticity for In-App Aftermarket services can now be calculated at [redacted], which implies a but-for price-cost margin of [redacted] percent, as seen in Table 5 above. This competitive price-cost margin is well within the range of AT&T's price-cost margins

---

473. Kahai et al., *supra*, at 510. This reflects AT&T's output-based market share. Its asset-based market share was even lower, at approximately 40 percent. *Id.*

474. Yang, *supra,* at 1417.

475. J.P. Morgan, *E-commerce Payments Trends: United States* (2019), *available at* www.jpmorgan.com/merchant-services/insights/reports/united-states.

476. The remainder was accounted for by bank transfers and other methods. J.P. Morgan, *2020 E-commerce Payments Trends Report: US*, *available at* www.jpmorgan.com/merchant-services/insights/reports/united-states-2020. *See also* D. Tighe, *Distribution of e-commerce payment methods in the United States in 2020*, STATISTA, *available at* www.statista.com/statistics/935676/payment-methods-used-for-online-transactions-usa/ (showing credit cards at 30 percent of e-commerce payments, debit cards at 21 percent, and digital wallets at 30 percent).

477. *See, e.g.,* Lewis Krauskopf, *Swiping their way higher: Visa, Mastercard could be the next $1 trillion companies*, REUTERS (January 31, 2020), *available at* www.reuters.com/article/us-visa-mastercard-stocks/swiping-their-way-higher-visa-mastercard-could-be-the-next-1-trillion-companies-idUSKBN1ZU0JA ("Visa holds a 60% share of the credit and debit card market[.]"). *See also* Julija A., *US Credit Card Market Share: Facts and Statistics,* FORTUNLY (November 23, 2021), *available at* fortunly.com/articles/credit-card-market-share/.

478. *See, e.g.,* Douglas Karr, *PayPal Market Share Statistics And Its History of Dominating Online Payment Processing*, MARTECH ZONE (Aug. 3, 2020), *available at* martech.zone/paypal-statistics-online-payments/.

479. PayPal's overall online market share has been independently estimated at 14 percent. *See* Stephanie Chevalier, *Which form of payment do you use most often for online shopping?*, Statista, *available at* www.statista.com/statistics/448712/online-shopping-payment-method-preference-usa/. *See also* Douglas Karr, *PayPal Market Share Statistics And Its History of Dominating Online Payment Processing*, MARTECH (Aug. 3, 2020), *available at* martech.zone/paypal-statistics-online-payments/ ("18% of all e-commerce is processed by PayPal[.]").

480. Kahai et al. at 508.

after entry by long-distance competitors (between [redacted] and [redacted] percent),[481] and [redacted] Alcoa's post-entry price cost margins of 12 percent.[482] Google's price to developers would fall to [redacted] per transaction in such a competitive but-for world, resulting in total savings of [redacted] per transaction relative to the actual world. In Part V.D below, I estimate that developers would pass on approximately [redacted] percent of these savings to consumers; accordingly, aggregate damages to consumers in the In-App Aftermarket come to [redacted] over the time period from 8/16/2016 through 12/31/2020. As seen above, Google's take rate would fall to [redacted] percent in this competitive but-for world, which would still afford Google a significant margin on the transactions in the In-App Aftermarket that it retains (Google's price-cost margin would be [redacted] percent, as shown in Row 15 of Table 5 above).

2. Analysis of Similar Platforms Corroborates My Competitive Take Rate In the In-App Aftermarket

221. It is common practice in digital markets outside of the In-App Aftermarket for entities to contract with outside payment processors. Epic considered an offer for payment processing from Codashop, which has partnered with numerous game developers across Southeast Asia,[483] [redacted][484] E-commerce apps that offer material, non-digital goods or services on Android phones outside the Play Store are not subject to Google's restrictions and use services such as Stripe, PayPal, and Square to process payments. These payment processors charge a materially lower commission to developers than Google. Table 6 provides a list of several prominent examples of take rates charged by other payment processors. The take rates shown in Table 6 reflect healthy competition among payment processors and are closer in magnitude to the implied costs associated with payment processing, a key component of the services that Google provides in the In-App Aftermarket.[485]

---

481. *Id.* at 510 ("The corresponding values of the Lerner index…are 0.29 and 0.13.").

482. Yang, *supra*, at 1417 ("Alcoa's residual demand elasticity is -8.3382. Then, the corresponding value of the Lerner index is 0.1199…").

483. *Coda Payments partners Riot Games for payments services across Southeast Asia,* THE PAYPERS, May 4, 2020, *available at* thepaypers.com/ecommerce/coda-payments-partners-riot-games-for-payments-services-across-southeast-asia--1242106.

484. [redacted]

485. As explained in the previous section, my analysis incorporates Google's financial information to account for the possibility that Google incurs additional marginal costs, beyond payment processing. Specifically, I conservatively include all direct costs recorded in GOOG-PLAY-000416245 (with the exception of content costs, which are irrelevant). Although many of the benchmark take rates in Table 6 entail a fee layered on top of a percentage of revenue, I do not impose such a fee in the but-for world modeled above.

TABLE 6: PAYMENT PROCESSORS AND THEIR TAKE RATES

	Payment Processor	Example Clients	Take Rate
(1)	PayPal	American Airlines, eBay, Facebook, Spotify	3.49% + $0.49
(2)	Stripe	Lyft, Under Armour, Blue Apron, Pinterest	2.9% + $0.30
(3)	Amazon Pay	Zuora, Shopify, BigCommerce, Magento	2.9% + $0.30
(4)	Braintree*	Uber, StubHub, Dropbox, Yelp	2.49% + $0.49
(5)	Square	Shake Shack, Postmates, Craver	2.6% + $0.10; 2.9% + $0.30†
(6)	Clover	Verizon Business	2.3% + $0.10
(7)	Authorize.net	TRX Cymbals, Prism Kites	2.9% + $0.30
(8)	Vanco	Churches and public schools	2.35% + $0.35; 2.75% + $0.45††
(9)	Fattmerchant	Lens Crafters, Jimmy Johns, Meineke, Maserati	$99 - $199/month + $0.06 - $0.15 per transaction†††
(10)	Adyen	Booking.com, McDonalds, Spotify, Microsoft	3.3% + $0.10; 3.95% + $0.12; 2.0% + $0.12 ††††
(11)	Google Pay**	Burger King, Dunkin Donuts, Target, Doordash	2.9%
(12)	Apple Pay	Best Buy, Taco Bell, Walgreens, Kohl's	3.0%

Notes: Take rates are based on fees for credit card usage. * Owned by PayPal. ** Not to be confused with Google Play Billing. † Square charges 2.6% + $0.10 for in-person swipes and 2.9% + $0.30 for online purchases. †† Vanco offers 2.75% + $0.45 with their "Start" plan ($10 monthly fee) and 2.35% + $0.35 for their "Sustain" plan ($49 monthly fee). ††† For the "Starter" plan, it costs $99 plus transactional fees running from $0.08 to $0.15; for the "Enterprise" plan, it costs $199 plus transactional fees running from $0.06 to $0.12. Fattmerchant claims this comes out to less than 1.5% for businesses that process more than $80K annually. *See* source below. †††† Adyen charges 3.3% + $0.10 for American Express, 3.95% + $0.12 for Discover, and interchange fees plus $0.12 for Mastercard and Visa. Interchange fees are 2% on average within the US. *See* www.adyen.com/blog/interchange-fees-explained.

Sources:

(1) Drew Strojny, *Stripe vs PayPal: Who should you choose?*, MEMBERFUL (Jan. 10, 2016), *available at* memberful.com/blog/stripe-vs-paypal/; Amanda Swan, *Which online stores accept PayPal?*, FINDER (Dec. 28, 2020), *available at* www.finder.com/paypal-store-list

(2) Drew Strojny, *Stripe vs PayPal: Who should you choose?*, MEMBERFUL (Jan. 10, 2016), *available at* memberful.com/blog/stripe-vs-paypal/; Amanda Swan ; Frank Kehl, *How Does Stripe Work? Everything You Need*

To Know About Processing Payments With Stripe, MERCHANT MAVERICK (July 22, 2021), *available at* www.merchantmaverick.com/how-does-stripe-work/
(3) Amazon Pay, *Innovative Merchant Payment Services for Small to Medium Businesses, available at* pay.amazon.com/business/small-business; Amazon Pay Sign up, *available at* pay.amazon.com/.
(4) Braintree, *Pricing*, *available at* www.braintreepayments.com/braintree-pricing; Braintree, *Boost Revenue with a Global Payments Partner, available at* www.braintreepayments.com/.
(5) Square, Payments, *available at* squareup.com/us/en/payments/pricing; Square, Payment Platforms, *available at* squareup.com/us/en/payments/payment-platform; Square, *Your all-in-one restaurant POS system, available at* squareup.com/us/en/point-of-sale/restaurants.
(6) Shannon Vissers, *How Much Does Clover Cost? Your Ultimate Guide To Clover Fees*, MERCHANT MAVERICK (Aug. 19, 2021), *available at* www.merchantmaverick.com/clover-pos-cost/ (This rate is for point-of-sale transaction; Clover charges 3.5% + $0.10 for online (keyed in) transactions); Verizon Communications, *Verizon Business offers touchless payment capability with Clover from Fiserv*, GLOBENEWSWIRE (Dec. 10, 2020), *available at* www.globenewswire.com/news-release/2020/12/10/2143226/0/en/Verizon-Business-offers-touchless-payment-capability-with-Clover-from-Fiserv.html/
(7) authorize.net, *Pricing, available at* www.authorize.net/sign-up/pricing.html; Featured Customers – Autorize.net, *available at* www.featuredcustomers.com/vendor/authorizenet/customers.
(8) Vanco, *Pricing, available at* www.vancopayments.com/egiving/pricing; Neal St. Anthony, *Vanco Payment Solutions grows as electronic bridge between churches, charities and their donors*, STAR TRIBUNE (Aug. 31, 2020), *available at* www.startribune.com/vanco-payment-solutions-grows-as-electronic-bridge-between-churches-charities-and-their-donors/572250712/.
(9) POSQuote.com, *Fattmerchant vs. Square: Which Payment Processor Is Best?, available at* posquote.com/review/fattmerchant-vs-square; Stax Payments – Healthcare, *available at* staxpayments.com/healthcare/.
(10) Adyen, *Pricing, available at* www.adyen.com/pricing; www.adyen.com/customers.
(11) Bankrate, *Guide to Google Pay* (Jan. 14, 2021), *available at* www.bankrate.com/finance/credit-cards/android-pay-google-pay-guide/#:~:text=While%20there%20is%20no%20charge,you%20use%20a%20credit%20card.; Google Pay, *For Business, available at* pay.google.com/about/business/partners/.
(12) Mark Jansen, Christian de Looper, and Paula Beaton, *PayPal vs. Google Pay vs. Venmo vs. Cash App vs. Apple Pay Cash*, DIGITAL TRENDS (July 5, 2021), *available at* www.digitaltrends.com/mobile/paypal-vs-google-wallet-vs-venmo-vs-square-cash/; MacRumors, *Apple Pay, available at* www.macrumors.com/roundup/apple-pay/#:~:text=Some%20of%20Apple's%20partners%20include,Bell%2C%20and%207%2D11.

D. Standard Economic Principles Show That All or Almost All Developers Would Pass Through to Consumers at Least a Portion of Any Savings from a Lower Take Rate

222. Google's take rate from developers typically ranges from 15 to 30 percent of revenue, with the average rate just below 30 percent.[486] In the competitive but-for world, these costs would have been lower. As explained in this section, standard economic models applied to data produced in discovery demonstrate all or almost all Class members would have benefitted as a result. I take no position on whether proof of pass-through is necessary under the law.

1. Standard Economics Shows That Prices Depend on Costs

223. One of the most universal principles of economics is that prices depend on costs.[487] Prices rise as marginal costs rise and fall as marginal costs fall. In perfectly competitive markets,

486. *See* Table 5, *supra*, Row 3 (showing Google's take rate for the In-App Aftermarket at [redacted] percent). Because the bulk of consumer expenditures in the In-App Aftermarket, Google's aggregate take rate across both markets is close to [redacted] percent.

487. *See, e.g.,* MANKIW, Chapter 4; Chapter 13.

firms pass through to buyers 100 percent of marginal cost increases or decreases in the form of correspondingly higher or lower prices.[488] In the absence of perfect competition, or indeed any competition, elementary economic principles of profit-maximization still dictate that prices will rise and fall with marginal costs.[489] Even if a firm has market or monopoly power, it will still maximize profits by passing costs to buyers; the fraction passed through depends on the shape of the demand curve (e.g., flat or curved). For example, with a linear demand curve (a downward sloping straight line), even in monopolistic markets, at least half of marginal cost savings are passed through to customers.[490] For nonlinear demand curves, the pass-through rate generally exceeds 50 percent.[491] For a demand curve with a constant elasticity, pass-through exceeds 100 percent.[492]

224. In markets served by competing firms with some degree of market power, such as developers on the Play Store or in the In-App Aftermarket, prices fall as marginal costs decline whenever firms choose price to maximize profit. A firm selling a single product (and thus facing downward-sloping demand) maximizes profit by charging a markup of price over cost equal to the inverse of the elasticity of demand.[493] This inverse elasticity rule can be written:

$$(P - C) / P = 1 / E_D,$$

where P is the price of the product, C is its marginal cost, and E_D is the firm's own-price elasticity of demand, defined as the percentage decrease in quantity demanded generated by a one percent increase in price. As seen above, the greater is C, the higher P must be to balance the equation.

225. Google's requirement that developers pay a percentage of their revenue to Google is mathematically equivalent to an increase in developers' marginal cost. To see this, let t be the take rate, where t is greater than zero and less than one, let Q be the quantity sold, and let $C(Q)$ be the developer's cost function.[494] The developer maximizes profit by maximizing the following expression:

$$\text{Profit} = \text{Total Revenue} - \text{Total Cost} = PQ(1 - t) - C(Q)$$

488. *Id.* at 272, Figure 1 (showing price = marginal revenue = marginal cost for a competitive firm).

489. *See, e.g.,* Jerry Hausman & Greg Leonard, *Efficiencies from the Consumer Viewpoint,* 17(3) GEORGE MASON LAW REVIEW 707, 708 (1999) ("profit maximization by the firm causes it to pass through at least some of the cost savings in terms of a lower price, even if the firm is a monopolist.").

490. *Id.* at 707 ("[S]o long as demand curves have the expected shape, the minimal amount of marginal cost savings passed on by a monopolist in terms of lower price is one-half of the cost savings.").

491. *Id.* at 721-724.

492. Theon van Dijk & Frank Verboven, *Quantification of Damages*, 3 ISSUES IN COMPETITION LAW AND POLICY 2331, 2342 (ABA Section of Antitrust Law 2008) ("When the price elasticity of demand is constant, η=0, and firms find it optimal to keep their percentage price-cost markup constant regardless of the cost conditions. This implies that a cost increase would lead to a higher absolute price cost-margin, which promotes pass-on.").

493. Hausman & Leonard, *supra*, at 713 (equation (5)). *See also* Landes & Posner at 937 (showing the analogous inverse-elasticity markup for a monopolist); Steven Berry, *Estimating Discrete Choice Models of Product Differentiation* 25(2) RAND JOURNAL OF ECONOMICS 242–262 (1994); Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects* 70 ANTITRUST LAW JOURNAL 257 (2002); Aviv Nevo, *Mergers with Differentiated Products: the Case of the Ready-to-Eat Cereal Industry*, 31(3) RAND JOURNAL OF ECONOMICS 395-421 (2000).

494. The cost function gives the developer's total cost as a function of the quantity produced. *See, e.g.,* MANKIW, *supra*, at 248-262.

Taking the derivative of the above equation with respect to Q, and setting the derivative equal to zero, the developer's profit-maximizing price satisfies the following equation:

$$(P - C^*) / P = 1 / E_D,$$

where $C^* = C/(1 - t)$.[495] Thus, when the developer pays a take rate to Google, the inverse-elasticity rule is modified as if the developer faced a higher marginal cost. For example, if the take rate is 30 percent, then $C^* = C/(1 - t) = C/(1 - 0.3) =$ [1.43 x C]. Put differently, charging a take rate of 30 percent is economically equivalent to a 43 percent increase in the developer's marginal cost.[496]

226. The competitive but-for world contemplates a long-run equilibrium, in which Google's take rate is substantially and permanently lower. Standard economics shows that a substantial and permanent reduction in developer's costs will cause substantial and permanent downward pricing pressure: Higher long-run costs mean higher prices, and vice-versa. In the long run, a profit-maximizing firm must either charge a price sufficient to cover its average *total* costs—inclusive of fixed costs—or exit the market.[497] To remain in business, a profit-maximizing firm must cover both its explicit costs of doing business and the opportunity costs incurred by not deploying its resources elsewhere.[498] Competition over the long run pushes prices downward towards levels that enable firms to cover their explicit costs and earn a competitive rate of return.[499] If Google's take rate were substantially and permanently lower, developers could cover all their costs and earn a competitive rate of return while charging consumers lower prices than they could otherwise.

2. Class Members Would Have Benefitted from Developers' Substantially Lower Costs through Various Economic Mechanisms

227. Class members would have benefitted from developers' lower costs in several ways. *First*, standard economics shows that Google's take rate influences a developer's initial decision regarding pricing for paid apps (and the pricing of any In-App Content) when the developer first enters the market (or when the developer first introduces new In-App Content). A developer faced with the prospect of paying up to 30 percent of its revenue to Google in perpetuity, all else equal, will need to charge a higher price to consumers than a developer facing a lower take rate. [redacted]

495. Profit maximization requires that the developer set marginal revenue (MR) equal to marginal cost (C). The developer's marginal revenue is $MR = (P + Q(\partial P/\partial Q))(1 - t)$, so profit maximization requires: $(P + Q(\partial P/\partial Q))(1 - t) = C$, which is equivalent to $(P + Q(\partial P/\partial Q)) = C^*$. Solving for the profit-maximizing markup, we obtain $P - P/E_D = C^*$, or $(P - C^*)/P = 1/E_D$.

496. The competitive but-for world contemplates a long-run equilibrium, in which Google's take rate is substantially and permanently lower. Marginal cost should therefore be interpreted as developers' long-run marginal cost, inclusive of the cost of inputs that are variable over the long run.

497. *See, e.g.,* MANKIW, *supra*, at 273-277.

498. *Id.* at 250-251; 279-284.

499. *Id.*

500. *See, e.g., ProtonMail wades into U.S. antitrust war*, AXIOS, July 22, 2021 ("At the top of Proton's list of grievances is the 30% commission Apple collects on subscriptions sold through its App Store, with Google planning to enforce the same fee (although Google recently announced a temporary extension to 2022). Proton's "freemium" model means it relies on paid subscriptions for revenue. The company raised prices for consumers to cover the Apple

[redacted] In addition, the price that the developer can charge to consumers will depend on the prices charged by competing developers: All else equal, a developer can charge a higher price when its competitors do the same—and competitors will charge higher prices when the take rate, and thus their costs, are higher.

228. *Second*, as explained above, standard economics shows that prices depend on costs; profit maximization dictates that decreased costs are passed through (at least partially) in the form of lower prices. In the competitive but-for world, developers' costs would have been substantially and permanently lower relative to the actual world. This, in turn, would have resulted in substantially and permanently lower prices paid by consumers.

229. *Third*, developers would face clear economic incentives to engage in steering in a competitive but-for world with more than one distribution channel by sharing a portion of the cost savings from a lower take rate with consumers who download Apps or In-App Content from a lower-cost platform. Prices for paid Apps and In-App Content are set by the developer. If a developer is charged a lower commission by one supplier relative to another, the developer can incentivize consumers to use the lower-cost supplier if the developer adjusts downward prices to consumers through the lower-cost source. This adjustment will steer customers to the favored supplier. In response, Google would be incentivized to lower its commissions from developers to prevent steering away from Google Play Billing (as well as from the Play Store in the Android App Distribution Market).

230. One possible mechanism for steering is illustrated by the Ultimatum Game, described originally by economist and Nobel laureate John Harsanyi in 1961.[502] Consider a setting in which a developer stands to save $1 in "service fees" per transaction if its customer elects to transact on a lower-cost platform than the Play Store for Apps or transacts through a payment processor other than Google Play Billing for In-App Content. To induce the consumer to select the lower-cost alternative, the developer must decide how much of the dollar to share with the consumer in the form of a reduced price for Apps or In-App Content. If the developer is not

---

fee, and is facing the prospect of doing the same when Google enforces its fee. "As a small company, there is no way we can afford to just absorb those fees," Miseviciute said. "So we are forced to raise the subscription prices for our consumers."). [redacted]

501. [redacted]

502. John Harsanyi, *On the Rationality Postulates underlying the Theory of Cooperative Games,* 5(2) JOURNAL OF CONFLICT RESOLUTION 179–196 (1961).

sufficiently generous with the amount offered, the consumer will continue to transact with Google, depriving the developer of any savings.

231. Google's documents explicitly contemplate steering as a consequence of lower take rates.[503] The slide below illustrates Google's projection of steering incentives if payment processing were unbundled, summarized by the asterisks in the final column. (It bears emphasis that, in the competitive but-for world, payment processing would be unbundled in tandem with distribution services in the In-App Aftermarket as Google's restrictions were eliminated.)

[redacted]

232. The first number in the "Option" column above displays Google's "unbundled" take rate—that is, the take rate excluding payment processing services. The second number is the extra amount that the developers would pay to Google, if Google also provided payment processing and if consumers chose Google rather than an alternative provider under Google's hypothetical scenarios. To illustrate, the third row displays [redacted] in the "Option" column, meaning that Google's hypothetical take rate consists of a base rate of [redacted] for Google's payment processing. The figure shows that as Google's unbundled take rate (again, excluding payment processing) decreases from [redacted], the greater are developers' steering incentives—that is, developers' incentives to "steer user choice" by lowering the price to consumers that select a lower-cost (non-Google) payment option. When the unbundled take rate is lower, there exists a greater likelihood that a cheaper alternative bundle can be

503. *See e.g.,* GOOG-PLAY-006829073 at GOOG-PLAY-006829085 (showing "Dev incentive to *steer user choice*" at lower take rates for payment processing) (emphasis added).

constructed, with a combined take rate (including payment processing) below 30 percent.[504] The "Consumer Spend w/Choice" column displays Google's estimate of how much revenue could potentially move to competitive billing systems under each option. For example, in the first row, there is [redacted], so the final column shows [redacted]

233. *Fourth*, even if one assumes that some developers would not lower their prices in the competitive but-for world, consumers still would benefit from quality improvements in Apps and In-App Content that developers would be able finance out of monies saved from lower take rates. Standard economics shows that competition drives firms to make competitive investments in product quality to keep pace with rivals.[505]

3. Statistical Analysis Using Standard Economic Models Confirms Widespread Pass-Through in a Competitive But-For World

234. In this section, I apply standard econometric models to data produced in discovery to estimate the extent to which developers would have passed lower costs (that is, lower take rates) to consumers in the form of lower prices in a competitive but-for world without Google's restrictions. My empirical analysis suggests very high pass-through rates among developers (on the order of 89.9 percent).

235. Using standard multiple-regression methods, I have econometrically estimated demand curves encompassing initial App downloads and purchases of In-App Content. In each regression, demand for a given App (or In-App Content) is modeled as a function of the price of that App (or the price of the In-App Content). All of the regressions include fixed effects.[506] Each fixed effect is unique to a given App, and to a purchase type for each transaction. Google's data presents three purchase types (initial downloads, in-app, and subscription). Thus, the regressions control for App-specific characteristics, as well as for differences in the demand for initial downloads of a given App, versus the demand for in-app purchases within that same App.

236. I have estimated nonlinear logit demand systems, which are commonly used in applied antitrust analysis.[507] In a logit demand system, the share for each product *j* is given by the following formula:

$$\ln(S_j / S_0) = \delta_j + \alpha P_j$$

504. If a developer does not use an alternative payment processor, Google's take rate remains at 30 percent. The developer can therefore lower its total take rate if the unbundled take rate (paid to Google) *plus* the cost of a third-party payment processor is below 30 percent. The lower the unbundled take rate, the easier it will be for a developer to construct such a bundle.

505. Department of Justice & Federal Trade Commission, *Horizontal Merger Guidelines* (2010), §10.

506. *See, e.g.,* JEFFREY WOOLDRIDGE, INTRODUCTORY ECONOMETRICS: A MODERN APPROACH, (THOMPSON 4TH ED. 2009), Chapter 14.1 [hereafter WOOLDRIDGE].

507. *See, e.g.,* Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects,* 70 ANTITRUST LAW JOURNAL 257 (2002). *See also* Jonas Bjornerstedt & Frank Verboven, *Merger simulation with nested logit demand*, 14(3) STATA JOURNAL 511-540 (2014).

Above, S_j is the share of product j, and S_0 is the share of the outside good—that is, the proportion of consumers that do not purchase any of the products.[508] The term δ_j represents factors other than price that shift share. These are modeled as fixed effects unique to a given App and purchase type (Initial Downloads, In-App, and Subscription).[509] Google's data presents three purchase types (Initial Downloads, In-App, and Subscription). Thus, the regressions control for differences in the demand for Initial Downloads of a given App, versus the demand for In-App purchases within that same App. In total, the regressions include over 200,000 control variables.

237. In Table 7 below, I econometrically estimate logit demand systems using standard ordinary least squares ("OLS") regressions.[510] In addition, I used standard instrumental-variable ("IV") regressions to correct for endogeneity.[511] In the IV regressions, taxes are used as an exogenous instrument that shifts price independently of other demand drivers.[512] The first two columns in Table 7 below report aggregate regressions that span all of Google's categories, with OLS regressions reported in the first column and IV regressions reported in the second column. Consistent with economic expectations, the regression results confirm a negative and highly statistically significant relationship between demand and price. For example, in Table 7 below, the coefficient on price (P) in Column (1) is -0.0107, and it is highly statistically significant, with a p-value of 0.000. According to Column (1), a one dollar increase in price leads to a 1.07 percent decrease in share.[513] Column (2) shows how the regression output changes after I correct for price endogeneity using IV. According to Column (2), a one dollar increase in price leads to a 5.16 percent decrease in share. This result is also highly statistically significant.

238. In Columns (3) and (4) of Table 7 below, I estimate separate logit demand systems for each of the categories used by Google, using a separate regression model for each category.[514] After correcting for price endogeneity using IV, the results for nearly all categories are consistent with economic expectations, in that the coefficient on price is both negative and statistically significant.[515] As summarized by the R-squared statistics below, the logit demand system regressions explain approximately 86 to 87 percent of the variation in the dependent variable.

---

508. Record evidence indicates that no more than [redacted] percent of users make purchases of In-App Content or initial downloads in a given month; this provides one possible estimate of the share of the outside good. GOOG-PLAY-000559379 at GOOG-PLAY-000559380 (a Play Store update for Alphabet Board (2020) stating "[t]oday, [redacted] of Play users make purchases on Play each month."). As explained below, because the pass-through rate in the logit model increases with the share of the outside good, I conservatively set it to zero for purposes of my pass-through calculations.

509. *See, e.g.,* WOOLDRIDGE, Chapter 14.1.

510. WOOLDRIDGE, *supra*, Chapter 3.

511. Instrumental variables techniques are used to identify the demand curve separately from the supply curve. *Id.* Chapter 15.

512. Taxes are used as instrumental variables because they directly affect price but are uncorrelated with other demand shifters. *Id.*

513. This represents a percent change, as opposed to percentage points.

514. In these regressions, the share of the outside good is calculated on a category-specific basis. For example, the share of the outside good for the "Art & Design" category is equal to the share of the market that either (1) made purchases in any of the other categories; or (2) did not purchase at all.

515. The only exception is the "Transportation" category, which accounts for less than 0.01 percent of consumer expenditures.

TABLE 7: ECONOMETRIC DEMAND CURVE ESTIMATES
LOGIT DEMAND: $\ln(S_j / S_0) = \delta_j + \alpha P_j$

		(1) OLS Aggregate	(2) IV Aggregate	(3) OLS By Category	(4) IV By Category
P	*All Categories*	-0.0107***	-0.0516***	N/A	N/A
		(0.000)	(0.000)	N/A	N/A
P	*Art & Design*	N/A	N/A	-0.00834***	-0.0233***
		N/A	N/A	(0.000)	(0.000)
P	*Auto & Vehicles*	N/A	N/A	0.00194**	-0.0141***
		N/A	N/A	(0.0438)	(0.000)
P	*Beauty*	N/A	N/A	-0.0108*	-0.103***
		N/A	N/A	(0.0656)	(0.000)
P	*Books & Ref*	N/A	N/A	-0.0329***	-0.0999***
		N/A	N/A	(0.000)	(0.000)
P	*Business*	N/A	N/A	-0.0116***	-0.0218***
		N/A	N/A	(0.000)	(0.000)
P	*Comics*	N/A	N/A	-0.0144***	-0.0319***
		N/A	N/A	(0.000)	(0.000)
P	*Communication*	N/A	N/A	-0.00745***	-0.0199***
		N/A	N/A	(0.000)	(0.000)
P	*Dating*	N/A	N/A	-0.00597***	-0.0314***
		N/A	N/A	(0.000)	(0.000)
P	*Education*	N/A	N/A	-0.0151***	-0.0395***
		N/A	N/A	(0.000)	(0.000)
P	*Entertainment*	N/A	N/A	-0.0258***	-0.0659***
		N/A	N/A	(0.000)	(0.000)
P	*Events*	N/A	N/A	-0.00595	-0.151***
		N/A	N/A	(0.584)	(0.000)
P	*Finance*	N/A	N/A	-0.0139***	-0.0408***
		N/A	N/A	(0.000)	(0.000)
P	*Food & Drink*	N/A	N/A	-0.00322***	-0.00514***
		N/A	N/A	(0.000)	(0.000)
P	*Game*	N/A	N/A	-0.00126***	-0.0709***
		N/A	N/A	(0.000)	(0.000)
P	*Health & Fitness*	N/A	N/A	-0.0203***	-0.0435***
		N/A	N/A	(0.000)	(0.000)
P	*House & Home*	N/A	N/A	-0.0153***	-0.0238***
		N/A	N/A	(0.000)	(0.000)
P	*Library & Demo*	N/A	N/A	-0.0308***	-0.204***
		N/A	N/A	(0.000)	(0.000)
P	*Lifestyle*	N/A	N/A	-0.0245***	-0.0546***
		N/A	N/A	(0.000)	(0.000)
P	*Maps & Nav*	N/A	N/A	-0.0230***	-0.0726***
		N/A	N/A	(0.000)	(0.000)
P	*Media & Video*	N/A	N/A	-0.0504	-0.138***
		N/A	N/A	(0.143)	(0.001)
P	*Medical*	N/A	N/A	-0.0122***	-0.0462***

		(1) OLS Aggregate	(2) IV Aggregate	(3) OLS By Category	(4) IV By Category
		N/A	N/A	(0.000)	(0.000)
P	*Music & Audio*	N/A	N/A	-0.0431***	-0.142***
		N/A	N/A	(0.000)	(0.000)
P	*News & Mag*	N/A	N/A	-0.00716***	-0.0202***
		N/A	N/A	(0.000)	(0.000)
P	*Parenting*	N/A	N/A	-0.0395***	-0.114***
		N/A	N/A	(0.000)	(0.000)
P	*Personalization*	N/A	N/A	-0.0770***	-0.277***
		N/A	N/A	(0.000)	(0.000)
P	*Photography*	N/A	N/A	-0.0397***	-0.0762***
		N/A	N/A	(0.000)	(0.000)
P	*Productivity*	N/A	N/A	-0.0216***	-0.0596***
		N/A	N/A	(0.000)	(0.000)
P	*Shopping*	N/A	N/A	-0.0116***	-0.0389***
		N/A	N/A	(0.007)	(0.000)
P	*Social*	N/A	N/A	-0.0175***	-0.0569***
		N/A	N/A	(0.000)	(0.000)
P	*Sports*	N/A	N/A	-0.00938***	-0.0336***
		N/A	N/A	(0.000)	(0.000)
P	*Tools*	N/A	N/A	-0.0276***	-0.0777***
		N/A	N/A	(0.000)	(0.000)
P	*Transportation*	N/A	N/A	0.0714***	0.820***
		N/A	N/A	(0.009)	(0.000)
P	*Travel & Local*	N/A	N/A	-0.0149***	-0.0487***
		N/A	N/A	(0.000)	(0.000)
P	*Video Players*	N/A	N/A	-0.0274***	-0.0496***
		N/A	N/A	(0.000)	(0.000)
P	*Weather*	N/A	N/A	-0.0397***	-0.142***
		N/A	N/A	(0.000)	(0.000)
	Constant	-18.65***	-18.39***	N/A	N/A
		(0.000)	(0.000)	N/A	N/A
	Includes FE?	Y	Y	Y	Y
	Number of FE	212,925	212,925	212,925	212,925
	Observations	1,905,224	1,905,224	1,905,224	1,905,224
	R-Squared	87.3%	87.5%	85.9%	86.2%

p-values in parentheses. *** p<0.01, ** p<0.05, * p<0.1. Columns (3) and (4) reflect the output of 35 separate regressions. For these columns, the “Number of FE” and “Observations” are the total number of Fixed Effects and Observations used in all of the regressions. The *R*-squared statistics in columns (3) and (4) are average *R*-squared values across all regressions, weighted by number of observations.

239. In Table 8 below I calculate pass-through rates, which are the ratio of the dollar change in a developer’s profit-maximizing price resulting from a one-dollar change in marginal cost. The logit demand system yields a pass-through rate equal to $[M - Q_j]/M$, where M is the size

of the market—inclusive of the outside good—and Q_j is the quantity sold of a given product.[516] All else equal, the larger is the share of the outside good (S_0), the larger is M, and the larger is the pass-through rate. Because the pass-through rate increases with the share of the outside good, I conservatively set S_0 equal to zero for purposes of calculating pass-through using the logit demand system.[517] As seen in the first row of Table 8 below, the logit demand system yields an overall pass-through rate of 89.9 percent. This estimate is calculated as the weighted average across all categories in the final column, with each category receiving a weight proportional to the quantity of transactions in that category. (Thus, the "Game" category receives more weight than others).

240. In summary, I estimate that developers' profit-maximizing response to a one dollar decrease in costs would be to decrease their prices to consumers by approximately $0.90. This price decrease is profit-maximizing because the marginal benefit to developers of lowering the price in the competitive but-for world—increasing the demand for Apps or In-App Content—exceeds the marginal cost of meeting the increased demand. If developers were to reduce their prices by less, their profit in the competitive world would be lower.

516. *See, e.g.,* Nathan Miller, Marc Remer, & Gloria Sheu, *Using cost pass-through to calibrate demand*, 118 ECONOMICS LETTERS 451-454, 452-453 (2013) (Equation 6 provides the formula for inverse of the pass-through rate, multiplied by negative one, which is: $-M/[M - Q_j]$, where M is the market size and Q_j is the quantity of product j. The pass-through rate is obtained by multiplying the expression by negative one and inverting it, which yields $[M - Q_j]/M$).

517. In addition, I conservatively aggregate purchase quantities by developer.

TABLE 8: PASS-THROUGH RATES

Category	Pass-Through Rate
ALL	**89.9%**
ART_AND_DESIGN	67.3%
AUTO_AND_VEHICLES	82.4%
BEAUTY	25.0%
BOOKS_AND_REFERENCE	80.4%
BUSINESS	81.6%
COMICS	55.6%
COMMUNICATION	86.6%
DATING	82.3%
EDUCATION	92.1%
ENTERTAINMENT	83.6%
EVENTS	62.0%
FINANCE	69.9%
FOOD_AND_DRINK	77.3%
GAME	92.3%
HEALTH_AND_FITNESS	94.7%
HOUSE_AND_HOME	59.7%
LIBRARIES_AND_DEMO	57.9%
LIFESTYLE	67.9%
MAPS_AND_NAVIGATION	82.0%
MEDIA_AND_VIDEO	88.4%
MEDICAL	92.6%
MUSIC_AND_AUDIO	40.7%
NEWS_AND_MAGAZINES	85.9%
PARENTING	79.1%
PERSONALIZATION	90.3%
PHOTOGRAPHY	90.7%
PRODUCTIVITY	88.6%
SHOPPING	47.4%
SOCIAL	89.2%
SPORTS	80.6%
TOOLS	95.8%
TRANSPORTATION	91.8%
TRAVEL_AND_LOCAL	89.0%
VIDEO_PLAYERS	87.7%
WEATHER	79.5%

4. Analysis of Pass-Through in Alternative Settings Corroborates Developer Pass-Through

241. In this section, I provide evidence and examples of pass-through that support my econometric calculation of the pass-through rate.

a. Apps That Avoid In-App Aftermarket Restrictions

242. Historically, some developers using the Play Store, including Netflix, Spotify, and Tinder, have refused to compel their customers to use Google Play Billing for In-App Content by instead redirecting them to a web browser.[518] Some developers allow consumers to transact both within the Google Play Store and Google Play Billing as well as outside of them at a lower price, permitting an apples-to-apples comparison that provides evidence of pass-through. Bookedin, an appointment scheduling software App, points out that it "need[ed] to charge extra to compensate for additional fees Apple and Google charge Bookedin to sell our app in their store."[519] The price offered on the Bookedin website for its "Pro 1" membership is $24/month vs. $34.99/month for the App sold through the Play Store, offering a 31 percent discount for avoiding Google's 30 percent take rate.[520] Down Dog, a yoga and exercise app, charges $7.99 a month on its website vs. $9.99 for the App purchased through the Google Play Store, implying a discount of 20 percent.[521] Tinder, a popular dating App and the second-highest grossing App on the Google Play Store, has discounted the subscription prices on its website by ten percent, relative to the price of the same subscriptions when purchased within the Play Store.[522] As observed by Jared Sine, Chief Legal Officer of Match Group (the parent company of Tinder), Google's and Apple's commissions amounted to "$500 million dollars that could be going back into the pockets of everyday consumers or deployed to hire employees or invest in new innovations."[523]

243. Spotify, a popular music-streaming App, reluctantly allowed consumers to purchase its subscription music service from within Apple's App Store (where it did not negotiate a lower take rate). Spotify was forced to raise the cost of its premium subscription service within

518. Ron Amadeo, *Google announces crackdown on in-app billing, aimed at Netflix and Spotify*, ARS TECHNICA (Sept. 20, 2020) ("Today, Netflix and Spotify don't use Google's in-app billing and instead kick new accounts out to a Web browser, where the companies can use PayPal or direct credit card processing to dodge Google's 30-percent fees"); *see also* Kif Leswing, *Google to enforce 30% take from in-app purchases next year*, CNBC (Sept. 28, 2020), *available at* www.cnbc.com/2020/09/28/google-to-enforce-30percent-cut-on-in-app-purchases-next-year.html; Nick Statt, *Tinder is now bypassing the Play Store on Android to avoid Google's 30 percent cut*, THE VERGE (Jul. 19, 2019), *available at* www.theverge.com/2019/7/19/20701256/tinder-google-play-store-android-bypass-30-percent-cut-avoid-self-install.

519. Bookedin, *Why are the prices different in App Store/Google Play?* (accessed June 15, 2021), *available at* support.bookedin.com/hc/en-us/articles/360028446492-Why-are-the-prices-different-in-App-Store-Google-Play-

520. ($34.99 - $24.00) / ($34.99) ≈ 31 percent.

521. ($9.99 - $7.99) / ($9.99) = ≈ 20 percent.

522. Based on prices observed on Tinder website vs. through Google Play on an Android device. Tinder Plus costs $9.99/month on Play compared to $8.99/month on its website, implying a discount of ($9.99 - $8.99) / ($9.99) ≈ 10%. Tinder Gold costs $14.99/month on Play compared to $13.49/month on its website, implying a discount of ($14.99 - $13.49) / ($14.99) ≈ 10%. Tinder Platinum costs $19.99/month on Play compared to $17.99/month on its website, implying a discount of ($19.99 - $17.99) / ($19.99) ≈ 10%.

523. Testimony for the S. Judiciary Committee, Subcomm. on Competition Policy, Antitrust, and Consumer Rights, at 2 (Jared Sine, Chief Legal Officer, Match Group, Apr. 21, 2021) [hereafter Sine Testimony].

the App Store from $9.99 per month to $12.99 per month in 2014.[524] This $3 increase can be understood as Spotify's attempt to roughly equalize its margins between its service through the internet ($9.99) and through the Apple App Store ($12.99).[525] Per Horacio Gutiérrez, Head of Global Affairs and Chief Legal Officer for Spotify: "Spotify could not absorb the IAP [in-app product] tax without raising its prices, because a large component of its costs are the licensing fees paid to record labels and music publishers."[526] Gutiérrez pointed out that "[o]ne doesn't need a Ph.D. in economics to recognize that Apple is hurting consumers by forcing competitors either to charge higher prices or preventing competitors from communicating offers of discounts or other promotional offers."[527] A European Commission investigation into prices charged by music-streaming providers showed that Apple's 30 percent commission rate was typically passed through in full to consumers.[528] [redacted] Finally, YouTube's subscription service charges a higher price on the Apple App Store "due to Apple's 30% transaction fee[.]"[530] Table 9 summarizes the discussion on implied pass-through rates for major app developers.

---

524. Testimony for the S. Judiciary Committee, Subcomm. on Competition Policy, Antitrust, and Consumer Rights, at 8 (Horacio Gutierrez, Head of Global Affairs and Chief Legal Officer, Spotify, Apr. 21, 2021).

525. Celena Chong, *Spotify shows its iPhone users how to save $3 by avoiding Apple's App Store*, BUSINESS INSIDER (July 8, 2015), *available at* www.businessinsider.com/spotify-shows-users-how-to-save-3-by-avoiding-apple-app-store-2015-7 ("Both Spotify and Apple Music technically charge $9.99 a month, but subscriptions purchased through the iTunes App Store charges a 30 percent fee — causing Spotify to charge $12.99 for its premium service within iTunes to generate the same revenue."). Apple's 30 percent commission would be $3.90 on a price of $12.99. By raising its price by only $3, Spotify was passing through 77 percent ($3/$3.90) of the increase due to the imposition of Apple's take rate.

526. Testimony for the S. Judiciary Committee, Subcomm. on Competition Policy, Antitrust, and Consumer Rights, at n. 13 (Horacio Gutierrez, Head of Global Affairs and Chief Legal Officer, Spotify, Apr. 21, 2021).

527. *Id.* at 9.

528. Statement by Executive Vice-President Margrethe Vestager on the Statement of Objections sent to Apple on App Store rules for music streaming providers (Apr. 30, 2021) ("Our investigation showed that this fee was passed on to end users by raising prices, typically from 9.99 to 12.99 Euros.").

529. [redacted]

530. Feng Dep. at 292:4-293:9, referencing PX 518 (March 2018 email from Google's Samer Sayigh stating "YouTube uses Apple Billing on iOS. YT charges $12.99/month on iOS (vs. $9.99 on Android + web), due to Apple's 30% transaction fee[.])").

TABLE 9: PRICE COMPARISON OF APPS THROUGH WEBSITE VS. APP STORE

	App	Product	Website Price	App Store Price	Implied Pass-Through Rate*
1	Tinder	Gold Membership	$13.49/month	$14.99/month (Google Play)	33%
2	BookedIn	Professional	$24/month	$34.99/month (Google Play)	105%
3	Down Dog	Unlimited Access to All Down Dog Apps	$7.99/month	$9.99/month (Google Play)	67%
4	Spotify	Premium Subscription Service	$9.99/month	$12.99/month (Apple App Store)	77%
5	Tidal	Premium Subscription Service	$9.99/month	$12.99/month (Apple App Store)	77%
6	YouTube	Subscription Service	$9.99/month	$12.99/month (Apple App Store)	77%

*Pass-through rate = (app price - website price) / ($ app commission - $ website commission) using an assumed zero percent commission charged on the website. Assuming a zero percent commission on the website is conservative so this estimate reflects a lower bound. The calculations conservatively assume a 30 percent take rate paid by the developer to Google (or Apple). To the extent that a developer pays a lower take rate, that developer's pass-through rate would be understated.

Sources: (1 - Tinder) Publicly advertised price; (2 - BookedIn) Appointment Booking & Online Scheduling Software - Pricing – BookedIn, *available at* support.bookedin.com/hc/en-us/articles/360028446492-Why-are-the-prices-different-in-App-Store-Google-Play (3 - Down Dog) Publicly advertised price; (4 - Spotify) Testimony for the U.S. Senate Judiciary Committee, Subcommittee on Competition Policy, Antitrust, and Consumer Rights, Horacio Gutierrez, Head of Global Affairs and Chief Legal Officer, Spotify, April 21, 2021 at 8; (5-Tidal) Shahar Ziv, *Here's Why Your Apple App Store Purchases May Be A Ripoff*, FORBES, (July 8, 2020), *available at* www.forbes.com/sites/shaharziv/2020/07/08/heres-why-your-apple-app-store-purchases-may-be-a-ripoff/?sh=77c6e9872007 ; (6-YouTube) Feng Dep. 292:4-293:9, referencing PX 518.

b. Pass-Through of Sales Taxes and Digital Service Taxes

244. Google's take rate is economically analogous to a tax on developers. Elementary economics shows how taxes are passed on to buyers: Regardless of whether the tax is imposed directly on buyers or instead on sellers, as a matter of theory, buyers end up paying the same portion of the tax.[531] The imposition of state and local sales taxes on digital goods therefore provides a useful example for understanding pass-through in this case. Apps often serve customers across a number of different local tax jurisdictions. When a digital product is subject to a tax, this burden is typically passed through in full to the customer; it is not absorbed by the seller. Indeed,

531. *See, e.g.,* MANKIW, *supra*, at 120-127.

software APIs have been created to facilitate passing through the correct amount of tax based on the local jurisdiction and the product being sold.[532] Economically speaking, this arrangement is tantamount to a commission being imposed on a developer that is fully passed through to the consumer. As summarized by Spotify on its website:

> Some state and local governments may also require us to collect tax (e.g. Sales Tax) if Spotify undergoes marketing/promotional activities in the state or locality, or uses local sales agents or consultants. This fee is included at the point of the transaction, which is why you might see a slightly different price on your receipt to the rate that's advertised.[533]

Examples like this abound: Netflix,[534] Hulu,[535] Amazon,[536] and Google[537] all offer similar disclaimers on their websites regarding local sales taxes. As one press report summarizes, "If you live in one of the nearly 25 states that charge sales tax on digital goods or services you likely pay more for everything from downloaded music, e-books and ringtones to streaming TV shows and video."[538]

E. Google Could Also Respond to Greater Competition By Increasing Its Play Points Program For Consumers

245. The two-sided model for the Android App Distribution Market and the one-sided model for the In-App Aftermarket were used to determine competitive but-for take rates under the assumption that the locus of competition, absent the Challenged Conduct, would be on take rates. For example, the two-sided platform model assumes that Google's access charge to consumers was near zero (and actually negative) in both the actual and competitive but-for worlds. An alternative, plausible response to the elimination of Google's restrictions would be for Google to

---

532. *See, e.g.*, Jennifer Dunn, *Sales Tax by State: Should You Charge Sales Tax on Digital Products?* (Feb. 13, 2018), *available at* www.taxjar.com/blog/sales-tax-digital-products ("The TaxJar API allows you to assign a product tax code to the products you sell. If you assign the product tax code for digital goods to the digital products you sell, the TaxJar API automatically charges your customer in any state the right amount of sales tax depending on that state's applicable laws.").

533. Spotify, *Does the price for Premium include tax?, available at* support.spotify.com/us/article/sales-tax/.

534. Netflix, *Taxes on your Netflix membership, available at* help.netflix.com/en/node/50068#:~:text=The%20Netflix%20advertised%20price%20does,membership%20includes%20streaming%20and%20games ("The Netflix advertised price does not include sales tax. If sales tax applies, it is stated separately on your monthly invoice").

535. Hulu, *Why was I charged tax?, available at* help.hulu.com/s/article/charged-sales-tax#:~:text=Why%20was%20I%20charged%20tax%3F&text=In%20certain%20jurisdictions%2C%20Hulu%20is,that%20is%20assessing%20the%20tax ("In certain jurisdictions, Hulu is required to charge tax on our services in order to comply with your state and local laws. This is based on your billing address. When applicable, these taxes are collected by Hulu and are then remitted to the jurisdiction that is assessing the tax.").

536. Amazon, *Help & Customer Service – Tax on Amazon Prime*, *available at* www.amazon.com/gp/help/customer/display.html?nodeId=202036230 ("If you choose to continue, you'll automatically be charged for Amazon Prime plus any applicable taxes").

537. Google, *Tax rates and value-added tax (VAT) - Play Console Help, available at* support.google.com/googleplay/android-developer/answer/138000?hl=en ("In accordance with sales tax requirements, Google is responsible for determining, charging, and remitting sales tax for Google Play Store app and in-app purchases by customers in these states. Google will collect and remit sales tax to the appropriate tax authority, as applicable. You don't need to calculate and send sales tax separately for customers in these states. Even if you're not located in the United States, this treatment will still apply.").

538. Melanie Hicken, *Are you paying the iTunes tax?*, CNN MONEY (June 5, 2013), *available at* money.cnn.com/2013/06/05/pf/taxes/itunes-tax/.

increase its loyalty points program for consumers to encourage their use of the Play Store and Google Play Billing rather than using any other competing source of Apps or In-App Content. An increase in Google's loyalty points would have the effect of reducing prices for purchases of Apps and In-App Content, without any requirement that developers steer consumers via discounting. Indeed, this form of competition is how credit cards compete to retain customers (funded via interchange fees to merchants), and how global distribution systems compete to retain travel agents (funded by booking fees charged to airlines). In addition to serving as a means to compete for consumers, a subsidy to consumers by Google can also bring value to developers by encouraging consumer spending.[539]

246. Given the current lack of competition due to the Challenged Conduct, the size and scope of Google's Play Points program is fairly modest. Android users who signed up pre-2019 must opt in to Google Play Points.[540] In addition, not all Apps participated in Google Play Points.[541] According to one analyst, under the current configuration of the program, "the spend-to-earn ratio is so steep that you would have to spend a pretty unrealistic amount of money in the Google Play Store to get enough points to actually do anything."[542] Despite its modest size relative to Google's take rate, the structure of Play Points is a reasonable facsimile of what an expanded program might look like in a competitive but-for world. Google awards Play Points for: (1) any purchase; (2) participation in weekly promotions (essentially getting extra points for spending on particular top games); and (3) installing new Apps that Google selects.[543] The points can then be spent on (a) Play credits (money to buy games); (b) priced initial App downloads or In-App Content; or (c) discounts on In-App Content.[544] Consumers may also reach higher "tiers" as they accrue points, which gives them access to additional benefits.[545]

247. Record evidence reveals that Google committed to reducing App prices via enhancing its points program to combat the threat of platform entry, a threat that never fully emerged in the actual world. Google first introduced Play Points in Japan, when it faced competition from Amazon.[546] Google noted that high valued users ("HVUs"), who accounted for half of all spending, presented a "churn risk."[547] It further noted that "Competitive Android stores

---

539. GOOG-PLAY-002653782.R at GOOG-PLAY-002653792.R ("Developers are realizing significant value from Play Points").

540. Jonathan Jaehnig, *What Are Google Play Points and How Can You Use Them?*, MAKE USE OF, (Apr. 8, 2021), *available at* www.makeuseof.com/what-are-google-play-points/.

541. *Id.*

542. *Id.*

543. GOOG-PLAY-000518034.

544. *Id.*

545. *Id.*

546. Google introduced its Play Points program in Japan in September 2018. *See* C. Scott Brown*, Google Play Points rewards program is real and in Japan right now,* ANDROID AUTHORITY (Sept. 18. 2018), *available at* www.androidauthority.com/google-play-points-rewards-905387/.

547. GOOG-PLAY-001284083.R at GOOG-PLAY-001284086.R. A "churn rate" tells how many customers a company loses over a time period in percentage terms. The risk that a company will lose customers can be referred to as "churn risk." *See* Patrick Icasas, *Your Customer is a Churn Risk (And Here's How We Know),* Catalyst (July 12, 2021), *available at* catalyst.io/blog/your-customer-is-a-churn-risk-and-heres-how-we-know.

such as Amazon App Store in Japan have reached 2-10% of developer revenue, mainly by attracting Play HVUs with heavy discounts (20 – 50% off)."[548]

248. Google next deployed the Play Points program in response to the ONE Store's entry in Korea. The ONE Store offered "cashback events," which amount to 30 to 50 percent of total transactions inside certain gaming Apps,[549] making points (rather than the take rate) the locus of competition. Google planned to combat ONE Store's "lowering rev share to court developers" through Play Points.[550] Google also noted that Samsung, through its Galaxy app store, was "courting developers via low 20% rev share."[551] In response to this dual threat in South Korea,[552] Google enhanced its own points program, launching in April 2019 and enrolling "millions of members" in "a few weeks" upon launch.[553] By the end of 2020, Google was offering a Play Points subsidy worth approximately two percent of consumer spend in South Korea.[554] These examples indicate that Google could respond to the elimination of its anti-competitive restrictions with incentives and discounts to consumers.

249. In the United States, Google's Play Points program was more widely introduced alongside Project Hug as part of a larger effort to combat the threat of platform competition. In a document, Google strategized about what to do with [redacted], a holdout on Project Hug.[555] [redacted] wanted a reduced take rate in addition to the other incentives Google was offering.[556] Google discussed various options. One was to [redacted]. Another was to [redacted]. A third option was to [redacted] Under this third approach, "We should let [redacted] know that Google is willing to invest Play's entire 30% share back to users in order to keep the ecosystem safe & growing for all."[557] Put differently, to avoid competition by another app store, such as proposed by [redacted] Google cared so much about not having consumers leave the Play Store that Google was willing to [redacted]

250. Further, Google initially ceded to mobile carriers [redacted] percentage points of its 30 percent take rate (or [redacted] percent) to fend off platform competition. This suggests that Google would be willing to cede as much to consumers to fend off competition in a but-for world without Google's anticompetitive restrictions.

251. To model the effect of an increased subsidy to consumers through Play Points in a competitive but-for world, I once again use the Rochet-Tirole two-sided platform model. However,

548. GOOG-PLAY-001284083.R at GOOG-PLAY-001284086.R. Google also noted that Japan "might just be the start as Amazon (historically) tries different approaches in a market (usually for a couple of years) before scaling up and going global with services." GOOG-PLAY-000879195R.

549. *See* Kim Jung-Min & Chea Sarah, *One Store Gains Ground in Local Android App Market*, KOREA JOONANG DAILY (Dec. 2, 2020), *available at* koreajoongangdaily.joins.com/2020/12/02/business/industry/One-Store-app-market-Google/20201202175300439.html.

550. GOOG-PLAY-000953420.R at GOOG-PLAY-000953422.R.

551. *Id.* at GOOG-PLAY-000953437.R.

552. GOOG-PLAY-000302766 at GOOG-PLAY-000302864 (citing decline in new purchasers in South Korea as reason for launching loyalty program).

553. GOOG-PLAY-000518034 at GOOG-PLAY-000518045.

554. GOOG-PLAY-002653782.R at GOOG-PLAY-002653793.R. *See also* GOOG-PLAY-005708308.

555. GOOG-PLAY-007329029.

556. *Id.*

557. *Id.* at GOOG-PLAY-007329030.

I now solve for a negative transaction price or subsidy to consumers in response to platform competition, assuming that the but-for take rate remains fixed at its observed average value of 29.3 percent. Under the current program, Google's Play Points can be applied to both initial App downloads and purchases of In-App Content—that is, Google does not have two different point programs.[558] It is reasonable to assume the structure of the program would remain the same in a competitive world, albeit with a larger subsidy for consumers, as Google would want to incentivize users to continue purchasing through its platform. Accordingly, I estimate the model only once to obtain a single subsidy in both markets as Google offers now. In contrast, above I estimated the competitive but-for take rate in the Android App Distribution Market separately from the competitive but-for take rate in the In-App Aftermarket, as nothing requires Google to charge the same take rate in those separate markets. My use of the Rochet-Tirole model to estimate a competitive but-for subsidy that can be spent on initial paid App downloads or In-App Content should not be taken to mean that the In-App Aftermarket is a two-sided antitrust market nor that the two separate antitrust markets are suddenly unified. Put differently, that I use the two-sided model to estimate a *single* consumer subsidy across both initial paid App downloads and purchases of In-App Content—a subsidy model that Google uses today and might use in a competitive but-for world—has no bearing on whether the In-App Aftermarket is one-sided or two-sided.

252. Holding the take rate fixed at the observed monopoly level, a monopoly platform operator would maximize its profits by setting the buyer-side platform price P_B such that the following equation is satisfied:

(V.11)
$$\frac{P_B + tS - C}{S + P_B} = \frac{1}{\varepsilon_B}$$

where ε_B is the price elasticity of demand from buyers for paid App downloads or In-App Content.[559] In the presence of competition, the platform operator would maximize profits with respect to its residual demand curve (market demand net of demand that is competed away by rivals), yielding the competitive analogous expression:

(V.12)
$$\frac{P_B + tS - C}{S + P_B} = \frac{1}{\varepsilon_{OB}}$$

where ε_{OB} is the own-brand price elasticity of demand from buyers for paid App downloads or In-App Content.[560]

253. My sources and methods for obtaining the monopoly scenario inputs shown in Equation (V.11) are:

- P_B^M is equal to the price "charged" by Google to consumers for each transaction made on its platform in the monopoly scenario. Through its Play Point loyalty program and other promotions, Google effectively charges a small negative price to consumers. I compute the average value of this subsidy as the sum of all promotions paid by Google for transactions made in both the Android App Distribution Market and In-App Aftermarket, divided by

558. For this reason, the model can be applied whether there is one market, or whether there is a separate aftermarket.

559. Details of how Equation (V.11) is derived are provided in Appendix 3.

560. Details of how Equation (V.12) is derived are provided in Appendix 3.

the total quantity of paid Apps downloaded in the Android App Distribution Market and purchases of In-App Content in the In-App Aftermarket, as observed in Google's transaction records.

- t is equal to the observed take rate, computed as the sum of all revenue retained by Google in both the Android App Distribution Market and In-App Aftermarket divided by the sum of total revenue spent by consumers in both the Android App Distribution Market and In-App Aftermarket (prior to Google's promotional expenditures, which are captured by P_B^M).

- S is equal to the average price charged for paid Apps in the Android App Distribution Market and In-App Content in the In-App Aftermarket in the monopoly setting.[561] I calculate S as the total amount of revenue spent by consumers (prior to receiving promotions from Google) in both the Android App Distribution Market and In-App Aftermarket divided by the total quantity of paid Apps downloaded in the Android App Distribution Market and purchases of In-App Content in the In-App Aftermarket, as observed in Google's transaction records.

- Marginal cost C represents the incremental cost incurred by Google of executing a transaction in the Android App Distribution Market or In-App Aftermarket. I refer to Google's financial data to infer this value, which suggests that transaction fees, customer support, and other fees are equal to [redacted] percent of consumer expenditures.[562]

- ε_B^M is the buyer-side price elasticity of demand for paid Apps in the Android App Distribution Market and In-App Content in the In-App Aftermarket. ε_B^M reflects the change in the quantity demanded by consumers for Android App Distribution Market or In-App Aftermarket transactions associated with a change in the price of the App or in-App product $S^M + P_B^M$ (inclusive of the buyer-side platform price).[563] Given the other inputs to the monopoly model, the value of ε_B^M is implied by Equation (V.11).

I hold C fixed across the monopoly and competitive scenarios. Because I am modeling competition on the buyer-side consumer price, I hold the developer-side take rate t fixed between scenarios also. Holding t fixed implies no change in the (pre-subsidy) product price S between scenarios. My sources and methods for obtaining the remaining inputs to the competitive scenario expression shown in Equation (V.12) are:

- P_B^C is the competitive buyer-side price charged by Google for each transaction on its platform. Using the other inputs to the model, Equation (V.12) allows me to solve for P_B^C.

- ε_{OB}^C is the "own-brand" price elasticity of demand for paid Apps in the Android App Distribution Market and In-App Content in the In-App Aftermarket by consumers in the presence of competition. ε_{OB}^C reflects the change in the quantity demanded from consumers

561. Apps that are free to download and free In-App content have a zero price and are therefore excluded from the analysis.

562. *See* GOOG-PLAY-000416245 (includes all Direct COS & Direct OpEx).

563. ε_B^M (which reflects consumer sensitivity to the total product price including buyer-side platform price, $P_B^M + S^M$) differs from, but is related to, the take rate buyer elasticity $\varepsilon_{B,t}^M$ (which reflects consumer sensitivity to the take rate, effectuated via pass-through) that is referred to in Section V.A.3. Further description of these parameters can be found in Appendix 3.

for Android App Distribution Market and In-App Aftermarket transactions—from Google in particular, hence, "own-brand"—associated with a change in App prices. Relative to its monopolistic analogue, this parameter reflects a scenario where Google faces competition from rival platforms; as such, the parameter will be greater in magnitude, because the presence of platform competition allows easier defection from consumers in the presence of a product price increase. I draw from the economics literature empirical evidence of industries that have shifted from monopoly to competition. I conservatively estimate that the buyer-side price elasticity of demand faced by Google shifts from a value of [redacted] (in the monopoly setting, as calculated using Equation (V.11)) to [redacted] in the competitive setting. I arrive at [redacted] using the relation between own-brand elasticity and market demand elasticity under the conservative assumption that Google maintains a 60 percent share of the Android App Distribution Market with an inelastic supply response from Google's rivals.[564] Further description of this input is included in Appendix 3.

Table 10 summarizes the inputs and resulting buyer-side platform price. Table 10 shows calculations made with respect to transactions in both the Android App Distribution Market and the In-App Aftermarket combined; the inputs will therefore vary from those used in Table 3. According to this model, the Play Points program would be expanded to be worth an average of $0.77 per transaction, or approximately 8.7 percent of consumer spend (in the competitive but-for world).[565] Because the expanded Play Points program is a direct subsidy to consumers, there is no need to estimate a pass-through model to establish antitrust impact.

---

564. Similar to Part V.C, *infra*, I use the relation $E_g = \frac{E_M}{S_g} + \frac{E_S(1-S_g)}{S_g}$ where E_g is Google's own-brand elasticity (reflecting price responses of both buyers and sellers), E_M is market elasticity, S_g is Google's market share, and E_S is the elasticity of supply of Google's rivals. I conservatively assume Google maintains a 60 percent market share and that $E_S = 0$. This implies that buyer price elasticity of demand changes from 5.30 in the monopoly setting (estimated using Equation V.12) to 8.83 = 5.30/0.6. *See, e.g.,* Landes & Posner.

565. Equal to the but-for buyer-side platform price of [redacted] divided by the product price of [redacted]

TABLE 10: BUT-FOR BUYER-SIDE PLATFORM COMPETITION MODEL, COMBINED ANDROID APP DISTRIBUTION AND IN-APP AFTERMARKET IMPACT AND DAMAGES (8/16/2016 – 12/31/2020)

Actual World (Monopoly, Eqn. (V.11))

#	Input	Description	Value	Source/Notes
[1]		Consumer Expenditure (US)	[redacted]	GOOG-PLAY-005535886 (US Consumers)
[2]		Google Revenue (US)		GOOG-PLAY-005535886 (US Consumers)
[3]		Google Promotional Expenditures (US)		GOOG-PLAY-005535886 (US Consumers)
[4]		Android App Distribution (Paid) and In-App Transactions		GOOG-PLAY-005535886 (US Consumers)
[5]=[1]/[4]	S	App Product Price		Calculated
[6]=[2]/[1]	t	Take Rate		Calculated
[7]=-[3]/[4]	P_B^M	Buyer-side Platform Price		Calculated
[8]=[5]+[7]	$S + P^M{}_B$	App Product Price Net of Promotions		Calculated
[9]	C	Marginal Cost		GOOG-PLAY-000416245
[10]	ε_B^M	Buyer Price Elasticity of Demand		Calculated (Eqn. (V.11))

But-For World (Competitive, Eqn. (V.12))

	Input	Description	Value	Source
[11]=[5]	S	App Product Price		Calculated (Eqn. (V.8))
[12]=[6]	t	Take Rate		GOOG-PLAY-005535886
[13]	P_B^C	Buyer-side Platform Price		Calculated (Eqn. (V.12))
[14]=[11]+[13]	$S + P^C{}_B$	App Product Price Net of Promotions		Calculated
[15]=[9]	C	Marginal Cost		GOOG-PLAY-000416245
[16]	ε_{OB}^C	Buyer Own-Price Elasticity of Demand		Economic theory/empirical studies
[17]=[8]-[14]		Consumer Savings Per Transaction		Calculated
[18]=[17]*[4]		**Aggregate Damages**		Calculated

254. For the purpose of comparing this result (which pertains to a two-sided market model of competition that occurs with respect to the buyer-side platform price) with results shown in Section V.B (which pertains to a two-sided market model of competition that occurs with respect to the take rate), it is useful to define a "net" take rate as the portion of consumer expenditures that Google retains, net of the amount given to consumers in the form of a subsidy. Google's but-for net take rate when platforms compete on subsidies in the combined Android App Distribution Market and In-App Aftermarket (equal to the [redacted] percent take rate less the 8.7 percentage point subsidy to consumers, effectively [redacted] percent) is slightly lower than Google's net take rate when platforms compete on take rates in the Android App Distribution Market (equal to [redacted] percent,

net of Play Points[566]). It bears noting that the dimension upon which to compete (take rate versus Play Points) is not solely Google's choice. If a rival app store were to compete by (say) offering app developers a 15 percent take rate, then Google could be forced to lower its take rate, in addition to enhancing its Play Points.[567] Because the precise means by which competition would unfold is hard to predict, I have elected to model both extremes here, and demonstrated that consumers would benefit under either and thus have been harmed by the Challenged Conduct.

255. As Table 10 shows, the resulting but-for average price of paid App downloads in the Android App Distribution Market and In-App Aftermarket is [redacted] down from the observed price of [redacted] (net of Google's promotional expenditures to consumers). This difference results in an average overcharge to consumers of [redacted] and aggregate damages of [redacted] (equal to [redacted] times [redacted] transactions) as a result of Google's restrictions, across the Class Period (August 16, 2016, through December 31, 2020).

256. In the actual world, Google awards Play Points to members of the program in proportion to their purchases made in the Play Store; any customer can enroll in Play Points free of charge. In the but-for world, awarding Play Points in proportion to all purchases made by Class members implies lower prices in the but-for world relative to the actual world, and therefore common impact on all or virtually all Class members.

F. A Rigid Pricing Structure Ensures That All Class Members Would Benefit From The Removal Of The Challenged Conduct

257. Google's commission structure affects all paid Apps and purchases of In-App Content by taking a fixed percentage. Moreover, while developers set different prices for their Apps and In-App Content, these differences do not vary by customer. Eliminating the Challenged Conduct would introduce competition driving the commission rate lower or resulting in further subsidies to consumers by enhancing loyalty programs. Enhanced loyalty programs again would not lead to differential prices across different consumers. Both lower commission rates and enhanced loyalty programs would therefore benefit all Class members.

258. This is true even when a Class member purchases from a developer that has received a discount off Google's standard take rate in the actual world. In the actual world, Google has imposed a headline take rate of 30 percent nearly universally, with the following exceptions: (1) a 15 percent take rate rolled out in 2018 for subscription apps (initially only for subscribers after their first year; more recently for all subscribers); (2) a 15 percent take rate charged for the first $1 million of sales (starting July 2021); and (3) instances where developers were able to negotiate a lower take rate. Nevertheless, Google's overall take rate has been at or very close to 30 percent for the vast majority of consumer expenditures: Google's own data show that Google has collected commissions in excess of [redacted] percent of consumer expenditures in both the Android

---

566. Drawing from the results shown in Table 3, the net take rate is equal to the but-for take rate (23.36%) minus the portion of consumer expenditures that are returned through Google's buyer-side platform price, ($-\frac{P_B^C}{S^C} =$ [redacted]).

567. In the competitive but-for world, rival app stores could compete on many dimensions, including lower take rates, a higher consumer point system, or exclusives with developers. *See, e.g.*, GOOG-PLAY-007329076 at GOOG-PLAY-007329084 (Google anticipating that Samsung would compete by entering into exclusive arrangements "with major Android game developers (e.g., $40M exclusives from Niantic)").

App Distribution Market and the In-App Aftermarket.[568] Within the "Game" category (by far the largest), over 99 percent of Apps have a take rate above [redacted] percent in the In-App Aftermarket.

259. To the extent that some developers received discounts from Google's 30 percent take rate in the actual world, such discounts would be similarly negotiated in the competitive but-for world.[569] Moreover, a Google witness has testified that although Google has offered lower take rates for "some programs for certain categories of developers…we don't, outside the context of those programs, wouldn't be negotiating it individually with developers on rev share."[570] This pricing structure, and Google's policy limiting negotiation of take rates with individual developers, implies class-wide impact, as all or almost all developers would pay a lower take rate in a more competitive world, and would pass on a portion of the savings to all or almost all Class members.[571] In Part VII below, I show how to compute damages for individual Class members based on their purchases; the data and inputs to these calculations are common to the Class.

G. Lower Prices Would Enhance Demand for Apps and In-App Content and Lower Take Rates Would Enhance the Supply of Apps and In-App Content Leading To Increased Output Relative to the Actual World

260. A foundational principle in economics is that "demand curves" are downward sloping—meaning that, all else equal, consumers will demand more of a product or service the lower its price.[572] How much more will be demanded depends on the consumer elasticity of the demand response to lower prices for Apps and In-App Content. As developers steer consumers to lower-cost app stores or payment processors in the but-for world via lower prices, consumers would respond by making more paid App downloads and purchases of In-App Content. This phenomenon can be understood graphically as a movement down the demand curve, resulting in higher output.

261. In a similar vein, the supply of Apps and In-App Content would increase as developers receive more for their Apps and In-App Content. Currently, developers receive only approximately 70 percent of the revenues generated from paid Apps and In-App Content in light of Google's take rate. Absent the Challenged Conduct, developers would realize larger proceeds, which would bring forward more App and In-App Content development, commensurate with a shifting out of the supply curve. Indeed, Google recognized that take-rate reductions, leading to higher revenues (price less commission) paid to developers, could increase output: In announcing its reduction in its commission to 15% for the first $1 million in revenue, Google's Vice President of Product Management explained that the new policy would provide "funds that can help

---

568. *See* Table 3, *supra*, Row 6; see also Table 5, *supra*, Row 3.

569. Hal J. Singer and Robert Kulick, *Class Certification in Antitrust Cases: An Economic Framework*, 17 GEORGE MASON LAW REVIEW (2010); Hal Singer, *Economic Evidence of Common Impact for Class Certification in Antitrust Cases: A Two-Step Analysis*, 25(3) ABA'S ANTITRUST (2011).

570. Rosenberg Dep. at 123:22-124:23 (further testifying that "We had – we had interest in advancing the ecosystem as a collective, and so that was a principle from the beginning").

571. To illustrate, suppose that Google's overall take rate falls from 30 percent in the actual world to 20 percent in the but-for world. A developer paying a 15 percent take rate in the actual world would pay a take rate of 10 percent in the but-for world (equal to 15*(20/30). This lower take rate would then be passed on to Class members in the form of lower prices.

572. *See, e.g.,* GEORGE STIGLER, THE THEORY OF PRICE 23 (McMillan 3rd ed. 1987) ("The 'demand curve' is the geometrical expression of the relationship between quantity purchased and price, and our law of demand says that demand curves have a negative slope.").

developers scale up at a critical phase of their growth by hiring more engineers, adding to their marketing staff, increasing server capacity, and more."[573]

262. There is no reason why the markets for Apps and In-App Content are any different from other markets where demand increases as prices fall. Indeed, my estimated elasticities of demand confirm this relationship. Accordingly, a reduction in the prices to consumers of Apps and In-App Content, which would result from a removal of Google's anti-competitive restrictions in the two relevant antitrust markets, would translate directly into enhanced demand for Apps and In-App Content. Given the digital nature of the products in both markets and thus the largely fixed cost of supplying both products[574]—the initial App and In-App Content—increased demand would translate into additional output for both products.

263. To show the output effect for each year in the Class Period from 2016 through 2020 (the latest available annual data), I begin with the aggregate commissions paid by developers to Google. The average actual take rates on paid downloads in the Android App Distribution Market was [redacted] percent and the average actual take rates on paid purchases of In-App Content was [redacted] percent. According to my two-sided market model, the average competitive take rate in the Android App Distribution Market would be 23.4 percent. According to my economic model, the average competitive take rate in the In-App Aftermarket would be 14.8 percent.

264. Developers' prices to consumers are influenced by the take rate and the pass-through rate. A pass-through rate of 89.9 percent (which I estimate in Section V.D.3 and use in my analysis in Sections V.B and V.C) implies a new average price in the Android App Distribution Market of $3.67 and a new average price in the In-App Aftermarket of $7.65, as shown in Tables 3 and 5, respectively. Consequently, Class members on average would have spent $0.30 less per transaction in the Android App Distribution Market and $1.34 less per transaction in the In-App Aftermarket.

265. To convert the price reductions into output effects, I apply a price elasticity of demand for initial App downloads and In-App Content of 1.5, using the estimate of the market-wide elasticity of demand in the In-App Aftermarket as shown in Table 5.[575] Applying these estimates to the average price declines for purchases of Apps and In-App Content yields a 11.5 percent increase in initial downloads of paid Apps and a 22.3 percent increase in purchases of In-App Content, holding elasticity constant.[576] Accordingly, the Challenged Conduct can be said to have suppressed output relative to competitive but-for-world levels.

---

573. Sameer Samat (Google Vice President, Product Management), *Boosting Developer Success on Google Play* (Mar. 16, 2021), *available at* android-developers.googleblog.com/2021/03/boosting-dev-success.html.

574. The marginal cost is the developers' marginal cost, plus the marginal cost of the record keeping, server hosting, auto updating, security, and authorization services that Google currently performs.

575. For the purpose of illustrating demand response I apply a single elasticity of 1.5 to each market, though separate elasticities for each market can be estimated. In addition, Ghose and Han (2014) estimate the elasticity of demand of Apps in the Android App Distribution Market at 3.7. *See* Anindya Ghose & Sang Pil Han, *Estimating Demand for Mobile Applications in the New Economy*, 60(6) MANAGEMENT SCIENCE 1470-1488 (2014).

576. In the Android App Distribution Market, an overcharge of $0.38 translates to a 9.6 percent decrease relative to the actual price of $3.97; the 9.6 percent decrease multiplied by an elasticity of 1.5 equals 14 percent. In the In-App Aftermarket, an overcharge of $1.34 translates to a 14.9 percent decrease relative to the actual price of $8.99; the 14.9 percent decrease multiplied by an elasticity of 1.5 equals 22 percent.

266. In addition to these direct output effects, the but-for world is likely to be characterized by increased innovation by software developers, which would redound to the benefit of consumers in the form of enhancements to quality, quantity, and consumer choice in both the Android App Distribution Market and In-App Aftermarket. For example, lower take rates would allow developers to reinvest to improve their products by "hiring more engineers, adding to their marketing staff, increasing server capacity, and more," just as Google recognized in launching its reduction in take rate for the first $1 million in developer revenue.[577] Record evidence indicates that Google was optimistic about the impact of a reduced take rate on developers, with one executive writing that "we're confident that this investment back into the developer community, particularly smaller ones, will lead to increased innovation, resulting in more choice and lower price for users."[578] Increased competition would also allow developers to meaningfully improve the quality of their payment solutions. Even YouTube, which used its own billing system instead of Google Play Billing, recognized that integrating Google Play Billing "limits innovation in relation to billing."[579]

267. Because the counterfactual experiment lies at the heart of antitrust analysis,[580] which involves comparing actual output against output but for the Challenged Conduct, output effects can occur even against a background of expanding output in the relevant market.[581] Output effects here take the form of fewer Apps downloaded and fewer purchases of In-App Content than would have otherwise occurred. Because the demand for such goods falls as their prices rise (that is, the demand curves slope downward), it follows that output contracted, even though in the In-App Aftermarket we observe output increasing over the Class Period. In other words, the but-for competitive output curves sit strictly above the actual output curves, illustrated in Figures 8 and 9 below.

577. Sameer Samat (Google Vice President, Product Management), *Boosting Developer Success on Google Play* (Mar. 16, 2021), *available at* android-developers.googleblog.com/2021/03/boosting-dev-success.html.

578. GOOG-PLAY-002358233 at GOOG-PLAY-002358236.

579. GOOG-PLAY-001088593, at GOOG-PLAY-001088596.

580. Theon van Dijk & Frank Verboven, *Quantification of Damages*, in 3 ISSUES IN COMPETITION LAW AND POLICY 2331, 2332 (ABA Section of Antitrust Law 2008) ("The difference between this counterfactual world and the actual world provides the measurement of damages").

581. John Newman, *The Output-Welfare Fallacy: A Modern Antitrust Paradox*, 107(2) IOWA L. REV. (2022 forthcoming) (explaining the "The trial court found_that_AmEx's no-steering rules had increased retail prices for nearly every consumer product sold in the United States (among other ill effects)," implying output effects per the appropriate counterfactual exercise.).

FIGURE 8: OUTPUT EFFECTS IN THE ANDROID APP DISTRIBUTION MARKET

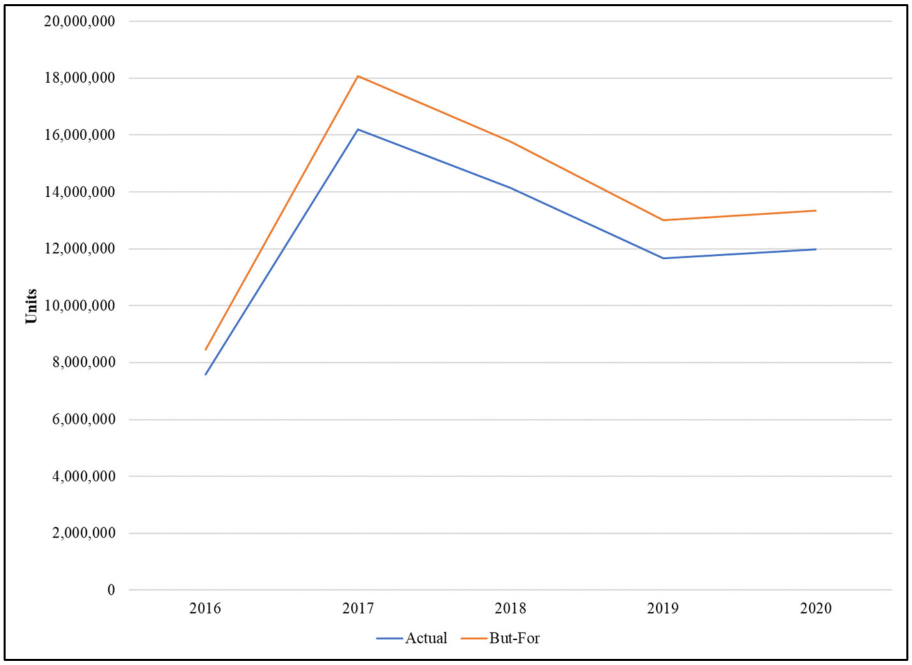
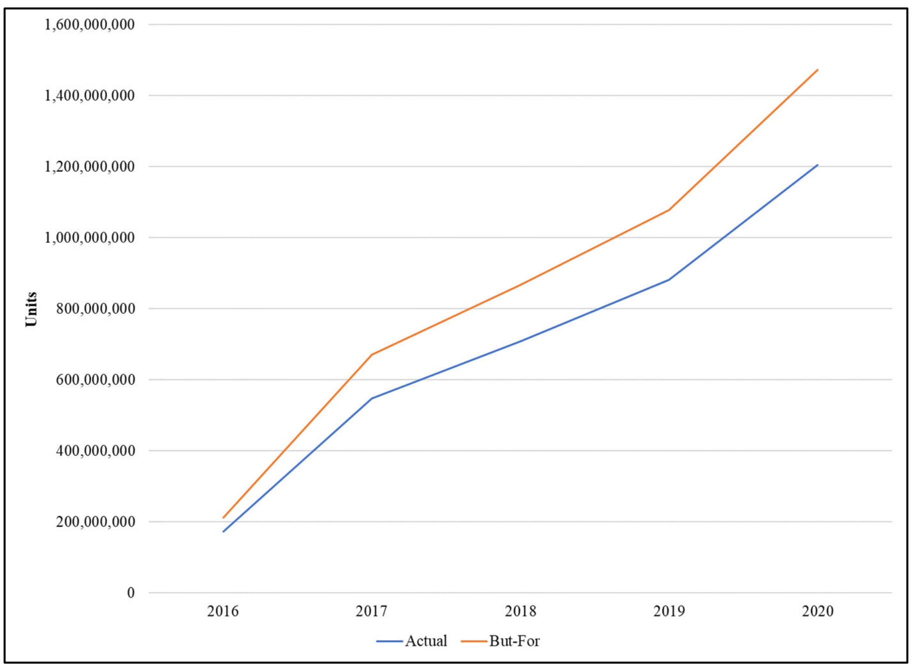


FIGURE 9: OUTPUT EFFECTS IN THE IN-APP AFTERMARKET



H. In the Competitive But-For World, Google Would Still Make a Profit from the Play Store

268. Eliminating Google's anticompetitive restraints would allow developers to use alternative app stores and be free from the first download onward to choose a payment processor and other suppliers of services that support the purchase of In-App Content. *Ex ante* competition

to be an alternative source of Apps or to become a default payment processor for a developer would result in competitive take rates. And lower take rates would redound to the benefit of consumers in the form of lower prices, as developers competed for the loyalties of consumers.[582]

269. Eliminating the Challenged Conduct would result in Google earning smaller margins relative to what it currently earns. One can solve for the implied Play Store profits in my competitive but-for world. For example, in 2020, excluding advertising revenue, and assuming Google preserves a 60 percent market share, the Play Store would have earned gross profit of [redacted] in the competitive but-for world. Assuming conservatively that operating expenses and infrastructure costs would have remained the same in the competitive but-for world, the Play Store still would have earned [redacted] in operating profit in the competitive but-for world (again excluding advertising). Once advertising is included, the Play Store's 2020 but-for profit is [redacted][583] Performing analogous calculations for 2021, the Play Store would have earned an estimated gross profit of [redacted] and operating profit of [redacted] in the competitive but-for world (again excluding advertising). Once advertising is included, the Play Store's 2021 but-for profit is estimated at [redacted].[584]

I. In the Absence of the Challenged Conduct, Google Would Refrain from Imposing a Fee on Consumers for Initial Downloads, Including on Free Apps

270. Because a large user base is critical, two-sided digital platform operators, such as Google, are often incentivized to provide free access to users or even to subsidize access to users.[585] This approach allows two-sided digital platforms to get "both sides on board." Encouraging use of the platform by one group may come at a cost to the platform operator, but it serves to attract the group on the opposite side of the platform.[586] In the instant setting, allowing consumers to browse the Play Store for free and download free apps creates a benefit to Google above the cost to create and maintain the app store due to the indirect network effects in attracting more developers and additional money Google can make by attracting advertisers. Google's network-driven incentive to "capture" as many users as possible by drawing them into the Google ecosystem (including the GMS suite) would not change in the absence of its anticompetitive conduct. Google would continue to benefit from indirect network effects even without its various restrictions—the more Apps it can attract, the more consumers will come to its platform—although its take rates on positively-priced Apps would be lower. However, any reduction in revenue from a lower take rate would be more than offset by the advertising revenue generated from maintaining its user base.

582. In the face of regulatory pressure, Google recently announced that it will it allow users in South Korea to use different in-app payment options, including in-app payment systems developed by app developers; Google will consequently decrease its take rate by four percentage points (from 15 percent to 11 percent). *See* Yoon Seon-hoon, *Google to abide by the 'forced in-app payment law'*, iNews24 (Nov. 4, 2021), *available at* n.news.naver.com/article/031/0000633296. In the competitive world contemplated here, Google would not be able to impose a take rate on in-app transactions unless Google were selected to be the payment processor by the developer.

583. GOOG-PLAY-000416245, GOOG-PLAY-001090227.

584. GOOG-PLAY-010801680, GOOG-PLAY-010801682.

585. *See, e.g.,* Thomas Eisenmann, Geoffrey Parker, and Marshall W. Van Alstyne, *Strategies for Two-Sided Markets*, HARVARD BUSINESS REVIEW 3-4, (2006) ("Because the number of subsidy-side users is crucial to developing strong network effects, the platform provider sets prices for that side below the level it would charge if it viewed the subsidy-side as an independent market."). *See also* Mark Armstrong & Julian Wright, *Two-sided markets, competitive bottlenecks and exclusive contracts*, 32 ECONOMIC THEORY 353-380, 359 (2007) ("If attracting one group (say buyers) makes the platform particularly attractive to the other group (say sellers), then buyers will be 'subsidized.'").

586. Rochet & Tirole at 991 (2003).

Google also would continue to obtain information about users, just as it does from its other apps in its GMS suite, which the company monetizes through the delivery of targeted ads to users.

271. Accordingly, Google likely would not impose even a modest fee on consumers for downloads of free Apps. Any attempt by Google to impose a transaction fee on consumers for free downloads would run counter to the company's basic business model and history to provide a widening array of free Apps or functionalities—such as Search, Maps, Gmail, You Tube, Chrome, and other Apps in its GMS suite—to consumers, but then collecting and monetizing information from consumers to realize and improve Google's targeted digital advertising. Imposing a fee on initial downloads would discourage consumers from downloading Apps through the Play Store, which in turn would jeopardize advertising revenues from the Play Store. Indeed, in 2020, Google earned [redacted] in the sale of ads that appeared in the Play Store, with almost all of those revenues falling to the bottom line.[587] Getting users on the Play Store lifts Google's advertising revenues above and beyond those earned on the Play Store, as those users are more inclined to remain in an Android environment, supporting other Google Apps. Moreover, the form-of-payment information, including a user's billing address, provides further personal identifying information for Google's advertising businesses. Advertising across the entire suite of Google's offerings accounts for over 80 percent of Google's total revenues.[588] By imposing a download fee on free Apps, Google would lose advertising revenue as some consumers ceased to use the Play Store and some advertisers would then not pay as much for ads or would stop advertising.[589]

272. Moreover, many Apps may be considered "experiential products,"[590] meaning that a consumer cannot discover its usefulness until they have downloaded the App and explored its functionalities. That consumers are accustomed to downloading Apps for free likely explains consumers' observed sensitivity to prices for paid Apps.[591] Imposing a fee on formerly free Apps would undermine this discovery process and thereby lower the value of the Play Store for consumers. Discouraging consumers from installing new Apps would also undermine the indirect network effects that Google is trying to harness—namely, a large customer base that attracts more developers to its platform and generates more advertising revenue. Google would still have an incentive to provide consumers access to free Apps (to try the Apps) because that is what sells the Apps and leads to purchases of paid In-App Content. In this respect, developers' and Google's incentives are aligned, competitive take rate or not. There is no reason to believe the business model of free initial App downloads would be eliminated because both Google and developers still want consumers to try and get hooked on Apps.

273. When Google cut its take rate from 30 percent to 15 percent for subscriptions longer than a year in 2018, or for all subscription revenue in 2022, it did not seek to offset the lost revenue by charging for Apps that were once free. Similarly, Google did not announce any increase in the

587. GOOG-PLAY-001090227.

588. Statista, *Advertising revenue of Google from 2001 to 2020*, *available at* www.statista.com/statistics/266249/advertising-revenue-of-google/ ("In 2020, Google's ad revenue amounted to 146.92 billion US dollars…Advertising accounts for the majority of Google's revenue, which amounted to a total of 181.69 billion U.S. dollars in 2020."). *See also* GOOG-PLAY-004113976 at GOOG-PLAY-004113979 (a 2015 presentation showing that "Core Ads" were responsible for 87 percent of total revenue).

589. Rosenberg Dep. at 410:8-413:14.

590. *See, e.g.*, Allison Kidd, *Technology experiences: what makes them compelling?*, HPLabs Technical Report (2001).

591. Anindya Ghose & Sang Pil Han, *Estimating Demand for Mobile Applications in the New Economy*, 60(6) MANAGEMENT SCIENCE 1470-1488 (2014) (estimating an elasticity of demand among users of the Play Store of -3.7).

take rate in the Android App Distribution Market in conjunction with its plans to offer a 15 percent take rate on the first $1 million in revenue for all developers in 2021. This evidence suggests that, in the competitive but-for world where Google would have to lower its take rate due to competition, it would still not seek to offset the lost revenue by charging consumers for downloading previously free Apps.

VI. ESTIMATION OF AGGREGATE DAMAGES CAN BE PERFORMED WITH COMMON METHODS AND EVIDENCE

274. Class members' aggregate damages are computed using the overcharges calculated in Parts V.B (Table 3) and V.C (Table 5). These damages are presented in Table 11. In Appendix 6, I break down these damages by U.S. state and territory.

TABLE 11: AGGREGATE DAMAGES SUMMARY, 8/16/2016 – 12/31/2020

	Android App Distribution Market (Table 3)	**In-App Aftermarket (Table 5)**	**Total**
[redacted]			

Notes: Total prices and overcharge are weighted averages across both markets. Quantities reflect paid transactions only.

275. Table 12 summarizes aggregate damages in the combined Android App Distribution Market and In-App Aftermarket under the modeling scenario where the locus of competition is on the consumer subsidy (as presented in Section V.E and Table 10), and under the combined markets take rate competition model, where competition occurs only with respect to the take rate in a single, combined market (as presented in Appendix 4 and Table A4).

TABLE 12: AGGREGATE DAMAGES SUMMARY, PLAY POINTS AND SINGLE MARKET TAKE RATE MODEL, 8/16/2016 – 12/31/2020

	Play Points Competition (Table 10)	**Take Rate Competition (Table A4)**
[redacted]		

Note: Quantities reflect paid transactions only.

VII. Estimation Of Class-Member-Specific Damages Can Be Performed With Common Methods And Evidence

276. I present a formula for computing a given Class member's damages based on the member's purchase history. While individual damages will depend on the individual purchases made by each consumer, the methodology for computing Class member-specific damages is common to the Class.[592]

277. An individual consumer's damages are equal to the quantity purchased of each product multiplied by the difference between the actual and but-for price paid for her purchases in the Android App Distribution Market and the In-App Aftermarket. Mathematically, individual damages can be expressed as:

$$\text{Damages} = \sum_i OC\_AD_i \times Q\_AD_i + \sum_j OC\_AM_j \times Q\_AM_j$$

where OC_AD_i represents the overcharge in the Android App Distribution Market (for paid apps)—that is, difference between the actual price and the but-for price. The term Q_AD_i represents the actual quantity purchased by a Class member in the Android App Distribution Market. Similarly, OC_AM_j represents the overcharge in the In-App Aftermarket (for In-App Content), and Q_AM_j represents the actual quantity purchased by a Class member in the In-App Aftermarket.[593]

278. The actual purchase quantities for each Class member (Q_AD_i and Q_AM_j) are reported in Google's transactional data. The overcharges are calculated from the economic models reviewed above. Specifically, OC_AD_i is calculated using the two-sided market model presented in Section V.B, and OC_AM_j is calculated using the one-sided market model presented in Section V.C.

279. The overcharges depend on (1) how much the take rate falls in the competitive but-for world; and (2) the extent to which the lower take rate is passed on in the form of lower prices. With respect to (2), I have calculated pass-through rates for different developer categories in Section V.D above. A Class member's damages will therefore depend on the category of apps that the customer purchased. All else equal, the higher is the pass-through rate of the category purchased by a particular Class member, the higher that Class member's damages will be.

280. With respect to (1), the drop in the take rate will often be similar or identical across categories: Most categories have similar (or identical) take rates in the actual world, and would also have similar (or identical) take rates in the but-for world. Nevertheless, I allow for variation in the but-for take rate across categories as follows: Suppose that the overall take rate is 30 percent in the actual world, and that a given category has an actual take rate of 29 percent. Suppose further that the overall but-for take rate is 20 percent. The category's but-for take rate would be calculated

592. For ease of exposition and presentation, here I present Class member-specific damages based on a Class member's purchases within each of Google's 35 categories. However, the same common framework could also be applied at the level of the developer. Class member-specific damages would then be calculated based on a Class member's expenditures at different developers, instead of different categories.

593. The summation operator (∑) is used because a Class member may have multiple transactions in the Android App Distribution Market and the In-App Aftermarket.

as [20 percent] x [29 percent]/[30 percent] = 19.33 percent. More generally, a category's but-for take rate is permitted to deviate from the overall but-for take rate by the same proportion in which the category's actual take rate deviates from the average actual take rate.

281. To illustrate, suppose that a consumer spent $10 on Apps in the "Art and Design" category in the Android App Distribution Market, as illustrated in the first row of Table 13 below. The actual take rate for the "Art and Design" category is [redacted] percent, and the but-for take rate for that same category is 23.2 percent. The pass-through rate for "Art and Design" is 67 percent. Given these inputs, the Class member's expenditures would fall to $9.47 in the but-for world. To see this, let P_b represent the consumer's but-for expenditures. Let t_a and t_b represent the actual and but-for take rates, and let γ be the pass-through rate. The difference between the Class member's actual and but-for expenditures is [$10 - P_b]. This difference satisfies the following equation:

$$[\$10 - P_b] = \gamma[\$10t_a - P_bt_b].$$

282. Solving for P_b, we obtain $P_b = \$10x([1- \gamma t_a]/[1 - \gamma t_b])$. Inserting the inputs from the "Art and Design" category, we have $P_b = \$10x([1- 0.67*0.299]/[1 - 0.67*0.232]) = \$9.47$. Thus, a Class member with $10 of expenditures in the "Art and Design" category of the Android App Distribution Market would have damages of $10 - $9.47 = $0.53. In other words, damages would equal 5.3 percent of expenditures (equal to $0.53/$10). Comparable calculations are performed for each remaining category in subsequent rows of Table 13.

TABLE 13: ILLUSTRATIVE CLASS MEMBER DAMAGES: ANDROID APP DISTRIBUTION MARKET

Category	CM Expenditure	Actual Take Rate	But-For Take Rate	Pass-Through Rate	But-For Expenditure	CM Damages	CM Overcharge (%)
ART_AND_DESIGN	$10.00	29.9%	23.2%	67%	$9.47	**$0.53**	**5.3%**
AUTO_AND_VEHICLES	$10.00	29.9%	23.2%	82%	$9.32	**$0.68**	**6.8%**
BEAUTY	$10.00	30.0%	23.3%	25%	$9.82	**$0.18**	**1.8%**
BOOKS_AND_REFERENCE	$10.00	29.7%	23.0%	80%	$9.34	**$0.66**	**6.6%**
BUSINESS	$10.00	29.7%	23.0%	82%	$9.33	**$0.67**	**6.7%**
COMICS	$10.00	29.5%	22.9%	56%	$9.58	**$0.42**	**4.2%**
COMMUNICATION	$10.00	29.9%	23.2%	87%	$9.27	**$0.73**	**7.3%**
DATING	$10.00	29.9%	23.2%	82%	$9.32	**$0.68**	**6.8%**
EDUCATION	$10.00	29.8%	23.1%	92%	$9.22	**$0.78**	**7.8%**
ENTERTAINMENT	$10.00	29.8%	23.1%	84%	$9.31	**$0.69**	**6.9%**
EVENTS	$10.00	30.0%	23.3%	62%	$9.51	**$0.49**	**4.9%**
FINANCE	$10.00	29.9%	23.2%	70%	$9.44	**$0.56**	**5.6%**
FOOD_AND_DRINK	$10.00	29.9%	23.2%	77%	$9.37	**$0.63**	**6.3%**
GAME	$10.00	29.7%	23.1%	92%	$9.22	**$0.78**	**7.8%**
HEALTH_AND_FITNESS	$10.00	30.0%	23.3%	95%	$9.18	**$0.82**	**8.2%**
HOUSE_AND_HOME	$10.00	29.8%	23.2%	60%	$9.54	**$0.46**	**4.6%**
LIBRARIES_AND_DEMO	$10.00	29.8%	23.1%	58%	$9.55	**$0.45**	**4.5%**
LIFESTYLE	$10.00	29.9%	23.2%	68%	$9.46	**$0.54**	**5.4%**
MAPS_AND_NAVIGATION	$10.00	29.8%	23.1%	82%	$9.32	**$0.68**	**6.8%**
MEDIA_AND_VIDEO	$10.00	29.5%	22.9%	88%	$9.27	**$0.73**	**7.3%**
MEDICAL	$10.00	29.9%	23.2%	93%	$9.21	**$0.79**	**7.9%**
MUSIC_AND_AUDIO	$10.00	29.8%	23.2%	41%	$9.70	**$0.30**	**3.0%**
NEWS_AND_MAGAZINES	$10.00	29.8%	23.1%	86%	$9.29	**$0.71**	**7.1%**
PARENTING	$10.00	30.0%	23.3%	79%	$9.35	**$0.65**	**6.5%**
PERSONALIZATION	$10.00	29.9%	23.2%	90%	$9.23	**$0.77**	**7.7%**
PHOTOGRAPHY	$10.00	29.8%	23.1%	91%	$9.23	**$0.77**	**7.7%**
PRODUCTIVITY	$10.00	29.8%	23.1%	89%	$9.26	**$0.74**	**7.4%**
SHOPPING	$10.00	29.9%	23.2%	47%	$9.64	**$0.36**	**3.6%**
SOCIAL	$10.00	29.9%	23.2%	89%	$9.25	**$0.75**	**7.5%**
SPORTS	$10.00	29.9%	23.2%	81%	$9.34	**$0.66**	**6.6%**
TOOLS	$10.00	29.8%	23.1%	96%	$9.18	**$0.82**	**8.2%**
TRANSPORTATION	$10.00	29.9%	23.2%	92%	$9.22	**$0.78**	**7.8%**
TRAVEL_AND_LOCAL	$10.00	29.8%	23.1%	89%	$9.25	**$0.75**	**7.5%**
VIDEO_PLAYERS	$10.00	29.7%	23.0%	88%	$9.27	**$0.73**	**7.3%**
WEATHER	$10.00	29.9%	23.2%	79%	$9.35	**$0.65**	**6.5%**

283. Table 14 below performs comparable calculations for the In-App Aftermarket. For example, suppose that a Class member spent $10 on in-app purchases in the "Game" category. This category has an actual take rate of 30.0 percent, a but-for take rate of 15.2 percent, and a pass-through rate of 92 percent. Applying the prior formula, the Class member's but-for expenditures are $10x([1- 0.92*0.30]/ [1 - 0.92*0.152]) = $8.42. This yields damages of $10 - $8.42 = $1.58 for each $10 of Class member expenditures in the "Game" category of the In-App Aftermarket. In other words, damages would be 15.8 percent of expenditures (equal to $1.58/$10).

TABLE 14: ILLUSTRATIVE CLASS MEMBER DAMAGES: IN- APP AFTERMARKET

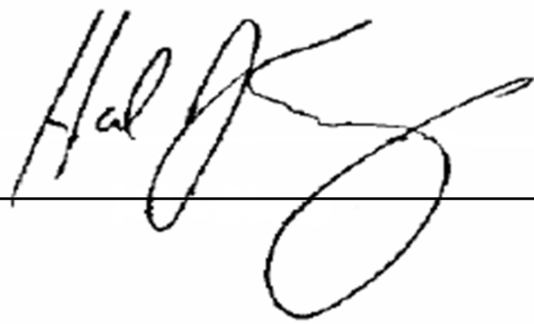
Category	CM Expenditure	Actual Take Rate	But-For Take Rate	Pass-Through Rate	But-For Expenditure	CM Damages	CM Overcharge (%)
ART_AND_DESIGN	$10.00	28.2%	14.3%	67%	$8.97	**$1.03**	**10.3%**
AUTO_AND_VEHICLES	$10.00	29.2%	14.8%	82%	$8.65	**$1.35**	**13.5%**
BEAUTY	$10.00	25.9%	13.2%	25%	$9.67	**$0.33**	**3.3%**
BOOKS_AND_REFERENCE	$10.00	29.5%	15.0%	80%	$8.67	**$1.33**	**13.3%**
BUSINESS	$10.00	27.6%	14.0%	82%	$8.75	**$1.25**	**12.5%**
COMICS	$10.00	29.6%	15.0%	56%	$9.12	**$0.88**	**8.8%**
COMMUNICATION	$10.00	28.9%	14.7%	87%	$8.59	**$1.41**	**14.1%**
DATING	$10.00	29.5%	15.0%	82%	$8.64	**$1.36**	**13.6%**
EDUCATION	$10.00	28.3%	14.4%	92%	$8.52	**$1.48**	**14.8%**
ENTERTAINMENT	$10.00	19.7%	10.0%	84%	$9.12	**$0.88**	**8.8%**
EVENTS	$10.00	29.9%	15.2%	62%	$9.00	**$1.00**	**10.0%**
FINANCE	$10.00	27.6%	14.0%	70%	$8.95	**$1.05**	**10.5%**
FOOD_AND_DRINK	$10.00	28.1%	14.3%	77%	$8.80	**$1.20**	**12.0%**
GAME	$10.00	30.0%	15.2%	92%	$8.42	**$1.58**	**15.8%**
HEALTH_AND_FITNESS	$10.00	27.3%	13.9%	95%	$8.53	**$1.47**	**14.7%**
HOUSE_AND_HOME	$10.00	28.5%	14.5%	60%	$9.08	**$0.92**	**9.2%**
LIBRARIES_AND_DEMO	$10.00	29.8%	15.2%	58%	$9.07	**$0.93**	**9.3%**
LIFESTYLE	$10.00	28.5%	14.5%	68%	$8.94	**$1.06**	**10.6%**
MAPS_AND_NAVIGATION	$10.00	28.5%	14.5%	82%	$8.69	**$1.31**	**13.1%**
MEDIA_AND_VIDEO	$10.00	29.8%	15.2%	88%	$8.50	**$1.50**	**15.0%**
MEDICAL	$10.00	27.5%	14.0%	93%	$8.56	**$1.44**	**14.4%**
MUSIC_AND_AUDIO	$10.00	19.1%	9.7%	41%	$9.60	**$0.40**	**4.0%**
NEWS_AND_MAGAZINES	$10.00	22.5%	11.4%	86%	$8.95	**$1.05**	**10.5%**
PARENTING	$10.00	28.1%	14.3%	79%	$8.77	**$1.23**	**12.3%**
PERSONALIZATION	$10.00	29.5%	15.0%	90%	$8.49	**$1.51**	**15.1%**
PHOTOGRAPHY	$10.00	28.1%	14.3%	91%	$8.56	**$1.44**	**14.4%**
PRODUCTIVITY	$10.00	26.7%	13.6%	89%	$8.68	**$1.32**	**13.2%**
SHOPPING	$10.00	29.7%	15.1%	47%	$9.26	**$0.74**	**7.4%**
SOCIAL	$10.00	29.7%	15.1%	89%	$8.49	**$1.51**	**15.1%**
SPORTS	$10.00	22.7%	11.6%	81%	$9.01	**$0.99**	**9.9%**
TOOLS	$10.00	27.7%	14.1%	96%	$8.49	**$1.51**	**15.1%**
TRANSPORTATION	$10.00	29.8%	15.1%	92%	$8.44	**$1.56**	**15.6%**
TRAVEL_AND_LOCAL	$10.00	28.2%	14.3%	89%	$8.59	**$1.41**	**14.1%**
VIDEO_PLAYERS	$10.00	28.5%	14.5%	88%	$8.59	**$1.41**	**14.1%**
WEATHER	$10.00	27.2%	13.8%	79%	$8.80	**$1.20**	**12.0%**

CONCLUSION

284. For the foregoing reasons, I conclude that the Challenged Conduct resulted in Class members overpaying for initial downloads from the Play Store and for the associated In-App Content.

* * *

Hal J. Singer, Ph.D.:



Executed on February 28, 2022.

APPENDIX 1: CURRICULUM VITAE OF HAL J. SINGER

Hal J. Singer

Econ One Research
Suite 510 805 15th St., N.W.
Washington, D.C. 20005
Phone: 202.312.3065
hsinger@econone.com

Education

Ph.D., The John Hopkins University, 1999; M.A. 1996, Economics

B.S., Tulane University, *magna cum laude*, 1994, Economics. Dean's Honor Scholar (full academic scholarship). Senior Scholar Prize in Economics.

Current Positions

ECON ONE, Washington, D.C.: Managing Director 2018-present.

GEORGETOWN UNIVERSITY, MCDONOUGH SCHOOL OF BUSINESS, Washington, D.C.: Adjunct Professor 2010, 2014, 2016, 2018, 2019, 2020, 2021, 2022.

GEORGE WASHINGTON UNIVERSITY, SCHOOL OF PUBLIC POLICY, GEORGE WASHINGTON INSTITUTE FOR PUBLIC POLICY, Washington, D.C.: Senior Fellow 2016-present.

Employment History

ECONOMISTS INCORPORATED, Washington, D.C.: Principal 2014-2018.

NAVIGANT ECONOMICS, Washington, D.C.: Managing Director, 2010-2013.

EMPIRIS, L.L.C., Washington, D.C.: Managing Partner and President, 2008-2010.

CRITERION ECONOMICS, L.L.C., Washington, D.C.: President, 2004-2008. Senior Vice President, 1999-2004.

LECG, INC., Washington, D.C.: Senior Economist, 1998-1999.

U.S. SECURITIES AND EXCHANGE COMMISSION, OFFICE OF ECONOMIC ANALYSIS, Washington, D.C.: Staff Economist, 1997-1998.

THE JOHNS HOPKINS UNIVERSITY, ECONOMICS DEPARTMENT, Baltimore: Teaching Assistant, 1996-1998.

Honors

Honoree, Outstanding Antitrust Litigation Achievement in Economics, American Antitrust Institute, *In re Lidoderm Antitrust Litigation,* Oct. 9, 2018.

Finalist, Outstanding Antitrust Litigation Achievement in Economics, American Antitrust Institute, *Tennis Channel v. Comcast*, Dec. 4, 2013.

Authored Books and Book Chapters

Do Municipal Broadband Networks Stimulate or Crowd Out Private Investment? An Empirical Analysis of Employment Effects, in THE IMPACT OF THE INTERNET ON JOBS (Lorenzo Pupillo, ed. Palgrave 2017).

THE NEED FOR SPEED: A NEW FRAMEWORK FOR TELECOMMUNICATIONS POLICY FOR THE 21ST CENTURY, co-authored with Robert Litan (Brookings Press 2013).

Net Neutrality Is Bad Broadband Regulation, co-authored with Robert Litan, in THE ECONOMISTS' VOICE 2.0: THE FINANCIAL CRISIS, HEALTH CARE REFORM AND MORE (Aaron Edlin and Joseph Stiglitz, eds., Columbia University Press 2012).

Valuing Life Settlements as a Real Option, co-authored with Joseph R. Mason, in LONGEVITY TRADING AND LIFE SETTLEMENTS (Vishaal Bhuyan ed., John Wiley & Sons 2009).

An Antitrust Analysis of the World Trade Organization's Decision in the U.S.-Mexico Arbitration on Telecommunications Services, co- authored with J. Gregory Sidak, in HANDBOOK OF TRANS-ATLANTIC ANTITRUST (Philip Marsden, ed. Edward Elgar 2006).

BROADBAND IN EUROPE: HOW BRUSSELS CAN WIRE THE INFORMATION SOCIETY, co-authored with Dan Maldoom, Richard Marsden and J. Gregory Sidak (Kluwer/Springer Press 2005).

Are Vertically Integrated DSL Providers Squeezing Unaffiliated ISPs (and Should We Care)?, co-authored with Robert W. Crandall, in ACCESS PRICING: THEORY, PRACTICE AND EMPIRICAL EVIDENCE (Justus Haucap and Ralf Dewenter eds., Elsevier Press 2005).

Journal Articles

Antitrust Anachronism: The Interracial Wealth Transfer in Collegiate Athletics Under the Consumer Welfare Standard, ANTITRUST BULLETIN (2021), co-authored with Ted Tatos.

Competing Approaches to Antitrust: An Application in the Payment Card Industry, 27(3) GEORGE MASON LAW REVIEW (2020), co-authored with Kevin Caves.

Understanding the Economics in the Dispute Between the Writers' Guild of America and the Big Four Talent Agencies, COMPETITION POLICY INTERNATIONAL ANTITRUST CHRONICLE (2020), co-authored with Ted Tatos.

Antitrust Out of Focus: The FTC's Myopic Pursuit of 1-800 Contacts' Trademark Settlements, ANTITRUST SOURCE (2019), co-authored with Geoff Manne and Josh Wright.

Countervailing Coordination Rights in the News Sector Are Good for the Public (A Response to Professor Yun), COMPETITION POLICY INTERNATIONAL ANTITRUST CHRONICLE (2019), co-authored with Sanjukta Paul.

When the Econometrician Shrugged: Identifying and Plugging Gaps in the Consumer Welfare Standard, 26 GEORGE MASON LAW REVIEW (2019), co-authored with Kevin Caves.

Applied Econometrics: When Can an Omitted Variable Invalidate a Regression?, ANTITRUST SOURCE (2017), co-authored with Kevin Caves.

Paid Prioritization and Zero Rating: Why Antitrust Cannot Reach the Part of Net Neutrality Everyone Is Concerned About, ANTITRUST SOURCE (2017).

The Curious Absence of Economic Analysis at the Federal Communications Commission: An Agency in Search of a Mission, INTERNATIONAL JOURNAL OF COMMUNICATIONS (2017), co-authored with Gerald Faulhaber and Augustus Urschel.

On the Utility of Surrogates for Rule of Reason Cases, COMPETITION POLICY INTERNATIONAL ANTITRUST CHRONICLE (2015), co-authored with Kevin Caves.

Analyzing High-Tech Employee: The Dos and Don'ts of Proving (and Disproving) Classwide Antitrust Impact in Wage Suppression Cases," ANTITRUST SOURCE (2015), co-authored with Kevin Caves.

Econometric Tests for Analyzing Common Impact, 26 RESEARCH IN LAW AND ECONOMICS (2014), co-authored with Kevin Caves.

Life After Comcast: The Economist's Obligation to Decompose Damages Across Theories of Harm, ANTITRUST (Spring 2014), co-authored with Kevin Caves.

Is the U.S. Government's Internet Policy Broken?, 5 POLICY AND INTERNET (2013), co-authored with Robert Hahn.

Avoiding Rent-Seeking in Secondary Market Spectrum Transactions, 65 FEDERAL COMMUNICATIONS LAW JOURNAL (2013), co-authored with Jeffrey Eisenach.

Vertical Integration in Multichannel Television Markets: A Study of Regional Sports Networks, 12(1) REVIEW OF NETWORK ECONOMICS (2013), co-authored with Kevin Caves and Chris Holt.

Assessing Bundled and Share-Based Loyalty Rebates: Application to the Pharmaceutical Industry, 8(4) JOURNAL OF COMPETITION LAW AND ECONOMICS (2012), co-authored with Kevin Caves.

Lessons from Kahneman's Thinking Fast and Slow: Does Behavioral Economics Have a Role in Antitrust Analysis?, ANTITRUST SOURCE (2012), co-authored with Andrew Card.

Assessing Competition in U.S. Wireless Markets: Review of the FCC's Competition Reports, 64 FEDERAL COMMUNICATIONS LAW JOURNAL (2012), co-authored with Gerald Faulhaber and Robert Hahn.

An Empirical Analysis of Aftermarket Transactions by Hospitals, 28 JOURNAL OF CONTEMPORARY HEALTH LAW AND POLICY (2011), co-authored with Robert Litan and Anna Birkenbach.

Economic Evidence of Common Impact for Class Certification in Antitrust Cases: A Two-Step Analysis, ANTITRUST (Summer 2011).

Addressing the Next Wave of Internet Regulation: Toward a Workable Principle for Nondiscrimination, 4 REGULATION & GOVERNANCE (2010), co-authored with Robert Hahn and Robert Litan.

Class Certification in Antitrust Cases: An Economic Framework, 17 GEORGE MASON LAW REVIEW (2010), co-authored with Robert Kulick.

The Economic Impact of Eliminating Preemption of State Consumer Protection Laws, 12 UNIVERSITY OF PENNSYLVANIA JOURNAL OF BUSINESS LAW 781 (2010), co-authored with Joseph R. Mason and Robert B. Kulick.

Net Neutrality Is Bad Broadband Regulation, THE ECONOMISTS' VOICE, Sept. 2010, co-authored with Robert Litan.

Why the iPhone Won't Last Forever and What the Government Should Do to Promote its Successor, 8 JOURNAL ON TELECOMMUNICATIONS AND HIGH TECHNOLOGY LAW 313 (2010), co-authored with Robert W. Hahn.

What Does an Economist Have to Say About the Calculation of Reasonable Royalties?, 14 INTELLECTUAL PROPERTY LAW BULLETIN 7 (2010), co-authored with Kyle Smith.

Is Greater Price Transparency Needed in the Medical Device Industry?, HEALTH AFFAIRS (2008), co-authored with Robert W. Hahn and Keith Klovers.

Evaluating Market Power with Two-Sided Demand and Preemptive Offers to Dissipate Monopoly Rent, 4 JOURNAL OF COMPETITION LAW & ECONOMICS (2008), co-authored with J. Gregory Sidak.

Assessing Bias in Patent Infringement Cases: A Review of International Trade Commission Decisions, 21 HARVARD JOURNAL OF LAW AND TECHNOLOGY (2008), co-authored with Robert W. Hahn.

The Effect of Incumbent Bidding in Set-Aside Auctions: An Analysis of Prices in the Closed and Open Segments of FCC Auction 35, 32 TELECOMMUNICATIONS POLICY JOURNAL (2008), co-authored with Peter Cramton and Allan Ingraham.

A Real-Option Approach to Valuing Life Settlement Transactions, 23 JOURNAL OF FINANCIAL TRANSFORMATION (2008), co-authored with Joseph R. Mason.

The Economics of Wireless Net Neutrality, 3 JOURNAL OF COMPETITION LAW AND ECONOMICS 399 (2007), co-authored with Robert W. Hahn and Robert E Litan.

Vertical Foreclosure in Video Programming Markets: Implication for Cable Operators, 3 REVIEW OF NETWORK ECONOMICS 348 (2007), co-authored with J. Gregory Sidak.

The Unintended Consequences of Net Neutrality, 5 JOURNAL ON TELECOMMUNICATIONS AND HIGH TECH LAW 533 (2007), co-authored with Robert E. Litan.

Does Video Delivered Over a Telephone Network Require a Cable Franchise?, 59 FEDERAL COMMUNICATIONS LAW JOURNAL 251 (2007), co-authored with Robert W. Crandall and J. Gregory Sidak.

The Competitive Effects of a Cable Television Operator's Refusal to Carry DSL Advertising, 2 JOURNAL OF COMPETITION LAW AND ECONOMICS 301 (2006).

Uberregulation without Economics: The World Trade Organization's Decision in the U.S.-Mexico Arbitration on Telecommunications Services, 57 FEDERAL COMMUNICATIONS LAW JOURNAL 1 (2004), co-authored with J. Gregory Sidak.

The Secondary Market for Life Insurance Policies: Uncovering Life Insurance's "Hidden" Value, 6 MARQUETTE ELDER'S ADVISOR 95 (2004), co-authored with Neil A. Doherty and Brian A. O'Dea.

Do Unbundling Policies Discourage CLEC Facilities-Based Investment?, 4 TOPICS IN ECONOMIC ANALYSIS AND POLICY (2004), co-authored with Robert W. Crandall and Allan T. Ingraham.

Foreign Investment Restrictions as Industrial Policy, 3 CANADIAN JOURNAL OF LAW AND TECHNOLOGY 19 (2004), co- authored with Robert W. Crandall.

Regulating the Secondary Market for Life Insurance Policies, 21 JOURNAL OF INSURANCE REGULATION 63 (2003), co- authored with Neil A. Doherty.

Interim Pricing of Local Loop Unbundling in Ireland: Epilogue, 4 JOURNAL OF NETWORK INDUSTRIES 119 (2003), co-authored with J. Gregory Sidak.

The Benefits of a Secondary Market for Life Insurance, 38 REAL PROPERTY, PROBATE AND TRUST JOURNAL 449 (2003), co- authored with Neil A. Doherty.

The Empirical Case Against Asymmetric Regulation of Broadband Internet Access, 17 BERKELEY TECHNOLOGY LAW JOURNAL 954 (2002), co-authored with Robert W. Crandall and J. Gregory Sidak.

How Can Regulators Set Nonarbitrary Interim Rates? The Case of Local Loop Unbundling in Ireland, 3 JOURNAL OF NETWORK INDUSTRIES 273 (2002), co-authored with J. Gregory Sidak.

Vertical Foreclosure in Broadband Access, 49 JOURNAL OF INDUSTRIAL ECONOMICS (2001) 299, co-authored with Daniel L. Rubinfeld.

Open Access to Broadband Networks: A Case Study of the AOL/Time Warner Merger, 16 BERKELEY TECHNOLOGY LAW JOURNAL 640 (2001), co-authored with Daniel L. Rubinfeld.

Cable Modems and DSL: Broadband Internet Access for Residential Customers, 91 AMERICAN ECONOMICS ASSOCIATION PAPERS AND PROCEEDINGS 302 (2001), co-authored with Jerry A. Hausman and J. Gregory Sidak.

Residential Demand for Broadband Telecommunications and Consumer Access to Unaffiliated Internet Content Providers, 18 YALE JOURNAL ON REGULATION 1 (2001), co-authored with Jerry A. Hausman and J. Gregory Sidak.

Determining the Source of Inter-License Synergies in Two-Way Paging Networks, 18 JOURNAL OF REGULATORY ECONOMICS 59 (2000).

A General Framework for Competitive Analysis in the Wireless Industry, 50 HASTINGS LAW REVIEW 1639 (2000), co- authored with J. Gregory Sidak and David Teece.

Capital Raising in Offshore Markets, 23 JOURNAL OF BUSINESS AND FINANCE 1181 (1999), co-authored with Ian Gray and Reena Aggarwal.

Expert Testimony Since 2012

Reviving Competition, Part 1: Proposals to Address Gatekeeper Power and Lower Barriers to Entry Online (U.S. House of Representatives Subcommittee on Antitrust)

Breaking the News – Journalism, Competition, and the Effects of Market Power on a Free Press (U.S. Senate Subcommittee on Competition Policy)

In Re: London Silver Fixing, Ltd. Antitrust Litigation, Case No. 1:14-md-02573-VEC (S.D. N.Y.)

In Re: JUUL Labs, Inc. Marketing, Sales Practices, and Products Liability Litigation, Case No. 19-md-02913-WHO (N.D. Ca.)

Paul Weidman et. al v. Ford Motor Company, Case No. 18-cv-12719 (E.D. Mich.)

Leinani Deslandes et al v. McDonald’s USA, LLC, Case No. 17-cv-04857 (N.D. IL)

In Re: Macbook Keyboard Litigation, Case No.: 5:18-cv-02813-EJD (N.D. Ca.)

Estate of Beverly Berland v. Lavastone Capital LLC, Case No. 1:18-cv-02002-CFC (D. Del.)

Donald Conrad et al. v. Jimmy John’s Franchise LLC, et al., No. 3:18-cv-00133-NJR (S.D. Ill.)

Zydus Pharmaceuticals Inc. and Cadila Healthcare Limited v. Takeda Pharmaceutical Company Limited et al., No. 18-01994 (FLW)(TJB) (D. N.J.)

In Re GSE Bonds Antitrust Litigation, No. 1:19-cv-01704-JSR (S.D. N.Y.)

beIN Sports, LLC v. Comcast Cable Communications, LLC, File No. CSR-8972-

P (FCC)

Chelsea Jensen, et al. v. Samsung Electronics et al., Court File No. T-809-18 (Federal Court in Canada)

Estate of Phyllis Malkin v. Wells Fargo Bank, N.A., No. 17-cv-23136 (S.D. Fl.)

In Re Capacitors Antitrust Litigation, Master File No. 3:14-cv-03264-JD (N.D. Ca.)

In re Foreign Exchange Benchmark Rates Antitrust Litigation, Case No. 1:13-cv-07789-LGS (S.D. N.Y.)

Massachusetts Technology Park Corporation v. Axia Netmedia Corporation, KCST USA, Inc., No. 01-17-0004-3049 (American Arbitration Association)

Cung Le et al. v. Zuffa, LLC, d/b/a Ultimate Fighting Championship and UFC, Case No.: 2:15-cv-01045-RFB-(PAL) (D. Nev.)

The Ohio State University v. New Par D/B/A Verizon Wireless, Case No. 2:15-cv-2866 (S.D. Oh.)

Authenticom, Inc. v. CDK Global, LLL; and The Reynolds And Reynolds Company, Case No. 17-cv-318 (W.D. Wis.)

Manmohan Dhillon et al. v. Anheuser-Busch, LLC et al. Case No. 14CECG03039 MBS (Cal. Fresno)

In re Lidoderm Antitrust Litigation, MDL Dkt. No. 14-md-02521-WHO (N.D. Cal.)

Maxon Hyundai Mazda et al. v. Carfax Inc., Case No. CV 2680 (AJN) (RLE) (S.D. N.Y.)

Philip R. Loy and Sharon Loy v. Womble Carlyle Sandridge & Rice, et al., Case No. 2014-cv-254012 (Ga. Super.)

In re MyFord Touch Consumer Litigation, Case No. 13-cv-3072-EMC (N.D. Cal.)

Sun Life Assurance Company of Canada v. U.S. Bank National Association, Case No. NO. 2:14-cv-04703-SJF-GRB (E.D. N.Y.)

Sun Life Assurance Company of Canada v. U.S. Bank National Association and Larry Bryan, Case No. 14-CIV-62610-BLOOM/VALLE (S.D. Fla.)

In the Matter of Flat Wireless, LLC, for and on behalf of its Operating Subsidiaries, v. Cellco Partnership d/b/a Verizon Wireless, and its Operating Subsidiaries, File No. EB-15-MD-005 (Federal Communications Commission)

Omni Healthcare et al. v. Health First Inc. et al, Case No. 6:13-CV-01509-RBD-DAB (M.D. Fla.)

Schuylkill Health System et al. v. Cardinal Health 200, LLC & Owens & Minor Distribution, Inc., Case No. 12-cv-07065-JS (E.D. Pa.)

Meda Pharmaceuticals Inc. v. Apotex, Inc and Apotex Corp., Case No. 01-14-0001-6315 (Am. Arbitration Ass'n)

Mark S. Wallach, et al v. Eaton Corporation, et al, Case No. 10-260-SLR (D. Del.)

STB Ex Parte No. 722 Railroad Revenue Adequacy (Surface Transportation Board)

In the Matter of 2014 Quadrennial Regulatory Review – Review of the Commission's Broadcast Ownership Rules and Other Rules Adopted Pursuant to Section 202 of the Telecommunications Act of 1996, MB Docket No. 14-50 (Federal Communications Commission)

Lindsay Kamakahi and Justine Levy, et al v. American Society for Reproductive Medicine and Society for Assisted Reproductive Technology, Case No.: 3:11-CV-1781 JCS (N.D. Cal.)

Salud Services, Inc. et al v. Caterpillar, Inc., Case No.: 1:12-cv-23927 (S.D. Fla.)

Gnanh Nora Krouch v. Wal-Mart Stores, Inc., Case No. CV-12-2217 (N.D. Cal.)

In the Matter of Petition for Rulemaking to Eliminate the Sports Blackout Rule, MB Docket No. 12-3 (Federal Communications Commission)

In the Matter of Review of Wholesale Services and Associated Policies, File No. 8663-C12-201313601 (Canadian Radio-Television and Telecommunications Commission)

Crafting a Successful Incentive Auction: Stakeholders' Perspectives (U.S. Senate, Committee on Commerce, Science, and Transportation)

Altergy Systems v. Enersys Delaware, Inc., Case No. 74-198-Y-001772-12 JMLE (American Arbitration Association)

In re New York City Bus Tour Antitrust Litigation, Master Case File No. 13-CV-07I1 (S.D. N.Y.)

SOCAN Tariff 22.A (Online Music Services, 2011-2013), CSI Online Music Services (2011-2013), SODRAC Tariff 6 - Online Music Services, Music Videos (2010-2013) (Copyright Board Canada)

Imperial Premium Finance, LLC, v. Sun Life Assurance Company of Canada (S.D. Fla.)

The Satellite Television Law: Repeal, Reauthorize, or Revise? (U.S. House of Representatives, Committee on Energy and Commerce)

Marchbanks Truck Service, et al. v. Comdata Network Inc., et al., Civil Action No. 07-1078-JKG (E.D. Pa.)

Patricia Reiter v. Mutual Credit Corporation, et al., Case No. 8:09-cv-0081 AG (RNBx) (C.D. Cal.)

In re Photochromic Lens Antitrust Litigation, MDL Docket No. 2173 (M.D. Fla.)

In the Matter of the Arbitration Between Washington Nationals Baseball Club v. TCR Sports Broadcasting Holdings, L.L.P. (Major League Baseball Revenue Sharing Definitions Committee)

Miguel V. Pro and Davis Landscape et al. v. Hertz Equipment Rental Corporation, No. 2:06-CV-3830 (DMC) (D.N.J.)

Game Show Network, LLC v. Cablevision Systems Corp., File No. CSR-8529-P (Federal Communications Commission)

Apotex, Inc., v. Cephalon, Inc., Barr Laboratories, Inc., Mylan Laboratories, Inc., Teva Pharmaceutical Industries, Ltd., Teva Pharmaceuticals USA, Inc., Ranbaxy Laboratories, Ltd., and Ranbaxy Pharmaceuticals, Inc., Case No. 2:06-cv-02768-MSG (E.D. Pa.)

In Re Airline Baggage Fee Antitrust Litigation, Civil Action No. 1:09-Md-2089-Tcb (N.D. Ga.)

Memberships

American Economics Association

American Bar Association Section of Antitrust Law

Reviewer

Journal of Risk and Insurance

Journal of Competition Law and Economics

Journal of Risk Management and Insurance Review

Journal of Regulatory Economics

Managerial and Decision Economics

Telecommunications Policy

APPENDIX 2: MATERIALS RELIED UPON

BATES DOCUMENTS

[redacted]
EPIC_GOOGLE_00123016
EPIC_GOOGLE_01389946
[redacted]
EPIC_GOOGLE_02075797
GOOG-PLAY-000416245
GOOG-PLAY-000000807
GOOG-PLAY-000001317
GOOG-PLAY-000001976
GOOG-PLAY-000003283.R
GOOG-PLAY-000004762.R
GOOG-PLAY-000005203
GOOG-PLAY-000005264
GOOG-PLAY-000025345
GOOG-PLAY-000053875
GOOG-PLAY-000053975
GOOG-PLAY-000054021
GOOG-PLAY-000054039
GOOG-PLAY-000054683
GOOG-PLAY-000127155
GOOG-PLAY-000131205.R
GOOG-PLAY-000225435
GOOG-PLAY-000226999
GOOG-PLAY-000229696
GOOG-PLAY-000231487
GOOG-PLAY-000233314
GOOG-PLAY-000237798
GOOG-PLAY-000258923
GOOG-PLAY-000259276
GOOG-PLAY-000259276
GOOG-PLAY-000292207.R
GOOG-PLAY-000297309.R
GOOG-PLAY-000302766
GOOG-PLAY-000338849
GOOG-PLAY-000367346.R
GOOG-PLAY-000379093
GOOG-PLAY-000400751.R

GOOG-PLAY-000416245
GOOG-PLAY-000416448
GOOG-PLAY-000416651
GOOG-PLAY-000416708
GOOG-PLAY-000417080
GOOG-PLAY-000439987.R
GOOG-PLAY-000443763
GOOG-PLAY-000443908.R
GOOG-PLAY-000445443.R
GOOG-PLAY-000446626
GOOG-PLAY-000464148
GOOG-PLAY-000518034
GOOG-PLAY-000542516.R
GOOG-PLAY-000556954
GOOG-PLAY-000559379
GOOG-PLAY-000572041.R
GOOG-PLAY-000620131
GOOG-PLAY-000620210
GOOG-PLAY-000620282
GOOG-PLAY-000620442
GOOG-PLAY-000620478
GOOG-PLAY-000620638
GOOG-PLAY-000620996
GOOG-PLAY-0006998204.R
GOOG-PLAY-000808062
GOOG-PLAY-000808375
GOOG-PLAY-000808433
GOOG-PLAY-000808451
GOOG-PLAY-000832219
GOOG-PLAY-000832471
GOOG-PLAY-000840773
GOOG-PLAY-000879194.R
GOOG-PLAY-000879195R
GOOG-PLAY-000880576.R
GOOG-PLAY-000932349
GOOG-PLAY-000953420.R
GOOG-PLAY-001055565
GOOG-PLAY-001075142
GOOG-PLAY-001076519
GOOG-PLAY-001088593
GOOG-PLAY-001090227
GOOG-PLAY-001143425
GOOG-PLAY-001184813
GOOG-PLAY-001284083.R
GOOG-PLAY-001291233
GOOG-PLAY-001317740

GOOG-PLAY-001337211
GOOG-PLAY-001381054
GOOG-PLAY-001381141
GOOG-PLAY-001385324
GOOG-PLAY-001400503
GOOG-PLAY-001404176
GOOG-PLAY-001423609
GOOG-PLAY-001438299
GOOG-PLAY-001449339
GOOG-PLAY-001451619
GOOG-PLAY-001455849
GOOG-PLAY-001508603
GOOG-PLAY-001559464.R
GOOG-PLAY-001559473
GOOG-PLAY-001745614
GOOG-PLAY-001847447
GOOG-PLAY-001877016
GOOG-PLAY-001882239
GOOG-PLAY-001886008.R
GOOG-PLAY-002011285
GOOG-PLAY-002076224.R
GOOG-PLAY-002358233
GOOG-PLAY-002416488
GOOG-PLAY-002604372
GOOG-PLAY-002653782.R
GOOG-PLAY-002891881
GOOG-PLAY-002994573
GOOG-PLAY-003332817.R
GOOG-PLAY-003334312
GOOG-PLAY-003335786
GOOG-PLAY-003604523
GOOG-PLAY-003604601
GOOG-PLAY-003604606
GOOG-PLAY-003605103
GOOG-PLAY-003776161.R
GOOG-PLAY-003881390.R
GOOG-PLAY-003894142
GOOG-PLAY-003899355.R
GOOG-PLAY-004113976
GOOG-PLAY-004119228.R
GOOG-PLAY-004136427
GOOG-PLAY-004144047.R
GOOG-PLAY-004146689.R
GOOG-PLAY-004147888
GOOG-PLAY-004235359
GOOG-PLAY-004237669.R

GOOG-PLAY-004253884
GOOG-PLAY-004278538
GOOG-PLAY-004338990
GOOG-PLAY-004470512
GOOG-PLAY-004486928.R
GOOG-PLAY-004489416.R
GOOG-PLAY-004494430.C
GOOG-PLAY-004499366
GOOG-PLAY-004502766
GOOG-PLAY-004508011
GOOG-PLAY-004509271
GOOG-PLAY-004541676
GOOG-PLAY-004542110
GOOG-PLAY-004552342
GOOG-PLAY-004556784
GOOG-PLAY-004559725.R
GOOG-PLAY-004564758
GOOG-PLAY-004630018.R
GOOG-PLAY-004703579
GOOG-PLAY-004714797
GOOG-PLAY-004716632
GOOG-PLAY-004717237
GOOG-PLAY-004721177
GOOG-PLAY-004904016
GOOG-PLAY-005027813
GOOG-PLAY-005535886
GOOG-PLAY-005535887
GOOG-PLAY-005653612.R
GOOG-PLAY-005696497
GOOG-PLAY-005705974
GOOG-PLAY-005706338
GOOG-PLAY-005706436
GOOG-PLAY-005706676
GOOG-PLAY-005706728
GOOG-PLAY-005708308
GOOG-PLAY-006355073
GOOG-PLAY-006359924
GOOG-PLAY-006381385
GOOG-PLAY-006817773.R
GOOG-PLAY-006829073.R
GOOG-PLAY-006990552
GOOG-PLAY-007038477
GOOG-PLAY-007038511
GOOG-PLAY-007203253
GOOG-PLAY-007264058
GOOG-PLAY-007315383

GOOG-PLAY-007317466
GOOG-PLAY-007317521
GOOG-PLAY-007317528
GOOG-PLAY-007317535
GOOG-PLAY-007317574
GOOG-PLAY-007317611
GOOG-PLAY-007329029
GOOG-PLAY-007329063
GOOG-PLAY-007329076
GOOG-PLAY-007617587
GOOG-PLAY-007657997
GOOG-PLAY-007745829
GOOG-PLAY-007819776
GOOG-PLAY-007847148
GOOG-PLAY-008006134
GOOG-PLAY-008111867
GOOG-PLAY-008384816
GOOG-PLAY-008427238
GOOG-PLAY-010801568
GOOG-PLAY-010801680
GOOG-PLAY-010801682
GOOG-PLAY2-000001992
GOOG-PLAY4-000284361
GOOG-PLAY4-000336290
GOOG-PLAY4-000339939
GOOG-PLAY4-002178046
GOOG-PLAY4-004677224
GOOG-PLAY4-007239946
[redacted]

Depositions

Deposition of Eric Chu (Dec. 20, 2021)

Deposition of Jamie Rosenberg (July 14, 2020)

Deposition of Jim Kolotorous (Feb. 2, 2022)

Deposition of Kevin Wang (Dec. 15, 2021)

Deposition of Lawrence Koh (Dec. 9, 2021)

Deposition of Paul Feng (Jan. 18, 2022)

Legal Documents

Cung Le et al. v. Zuffa, LLC d/b/a Ultimate Fighting Championship, Minute Entry, 2:15-cv-01045-RFB-BNW (D. Nev. Dec. 10, 2020), ECF No. 781

[redacted]

In re Delta/AirTran Baggage Fee Antitrust Litig., 317 F.R.D. 665 (N.D. Ga. 2016)

In re Lidoderm Antitrust Litig., No. 12-md-02521, 2017 WL 679367 (N.D. Cal. Feb. 21, 2017)

Johnson v. Arizona Hosp. and Healthcare Assoc. No. CV 07-1292-PHX-SRB, 2009 WL 5031334 (D. Ariz. July 14, 2009)

Meijer, Inc. v. Abbott Laboratories, No. C 07-5985 CW, 2008 WL 4065839 (N.D. Cal. Aug. 27, 2008)

Natchitoches Parish Hosp. Serv. Dist. v. Tyco Intl., Ltd., 262 F.R.D. 58 (D. Mass. 2008)

Newcal Indus., Inc. v. Ikon Office Sol., 513 F.3d 1038, 1049-50 (9th Cir. 2008)

Ohio v. American Express Co., 585 U.S. (2018), *available at* www.supremecourt.gov/opinions/17pdf/16-1454_5h26.pdf

Rule 52 Order After Trial On The Merits, Epic Games Inc. v. Apple, Inc., Case No. 4:20-cv-05640-YGR

Southeast Missouri Hospital and St. Francis Medical Center v. C.R. Bard, No. 1:07cv0031 TCM, 2008 WL 4372741 (E.D. Mo. Sept. 22, 2008)

United States v EI du Pont de Nemour and Co.

Weyerhaeuser Co. v. Ross-Simmons Hardwood Lumber Co., 549 U.S. 312 (2007)

Literature

Aaron Director & Edward H. Levi, *Law and the Future Trade Regulation*, 51 Nw. U. L. Rev. 281 (1956)

Aaron Edlin, Scott Hemphill, Herb Hovenkamp, and Carl Shapiro, *Activating Actavis*, Antitrust, (2013)

Agam Gupta et al., *Combating incumbency advantage of network effects: The role of entrant's decisions and consumer preferences*, 20(1) Competition and Regulation in Network Industries 3-32 (2019)

Anindya Ghose & Sang Pil Han, *Estimating Demand for Mobile Applications in the New Economy*, 60(6) Management Science 1470-1488 (2014)

Avi Goldfarb and Catherine Tucker, *Digital Economics*, 57 Journal of Economic Literature 3-43 (2019)

Aviv Nevo, *Mergers with Differentiated Products: the Case of the Ready-to-Eat Cereal Industry*, 31(3) RAND Journal of Economics 395-421 (2000)

Benjamin Edelman & Damien Geradin, *Android and competition law: exploring and assessing Google's practices in mobile*, 12(12-3) European Competition Journal 159-194 (2016)

Benjamin Edelman & Julian Wright, *Price Coherence and Excessive Intermediation,* 130 Q. J. Econ. 1283 (2015)

Brigitte C. Madrian & Dennis F. Shea, *The Power of Suggestion: Inertia in 401(k) Participation and Savings Behavior*, 116(4) Quarterly Journal of Economics 1149-1187 (2001)

Daniel O'Brien and Abraham Wickelgren, A critical analysis of critical loss analysis, 71 Antitrust L.J. (2003)

David Evans, *Two-Sided Market Definition in Market Definition In Antitrust: Theory And Case Studies* (ABA Section of Antitrust Law) 1-35 (2009), *available at* papers.ssrn.com/sol3/papers.cfm?abstract_id=1396751

Dennis Carlton, *A General Analysis of Exclusionary Conduct and Refusal to Deal—Why* Aspen *and* Kodak *Are Misguided*, 68(3) Antitrust Law Journal 659-983 (2001)

Einer Elhauge & Barry Nalebuff, *The Welfare Effects of Metering Ties*, 33 Journal of Law, Economics & Organization 68 (2017)

Einer Elhauge, *Tying, Bundled Discounts and the Death of Single Monopoly Profit Theory*, 123(2) Harvard Law Rev. 397-481 (2009)

Erik Hovenkamp, *Platform Antitrust*, 44 Journal of Corp. Law 713 (2019)

George Stigler, The Theory of Price (McMillan 3rd ed. 1987)

Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects* 70 Antitrust Law Journal 257 (2002)

Hal J. Singer and Robert Kulick, *Class Certification in Antitrust Cases: An Economic Framework*, 17 George Mason Law Review (2010)

Hal Singer & Andrew Card, *Lessons from Kahneman's Thinking Fast and Slow: Does Behavioral Economics Have a Role in Antitrust Analysis?*, Antitrust Source (2012), *available at*

www.semanticscholar.org/paper/Lessons-from-Kahneman-%E2%80%99-s-Thinking-%2C-Fast-and-Slow-Kahneman/8abf422dc2aca5adf6fe6c20e9064863f64819dd?p2df

Hal Singer, *Economic Evidence of Common Impact for Class Certification in Antitrust Cases: A Two-Step Analysis*, 25(3) ABA's Antitrust (2011)

Herbert Hovenkamp, *Digital Cluster Markets*, Columbia Business Law Review 1-30 (forthcoming 2022)

Jean-Charles Rochet & Jean Tirole, *Platform Competition in Two-Sided Markets*, 1(4) European Economic Association 990-1029 (2003)

Jeffrey Eisenach and Kevin Caves, *What Happens When Local Phone Service Is Deregulated?* Regulation 34-41 (2012)

Jeffrey M. Perloff, Microeconomics A-34 (Pearson 7th ed. 2015)

Jeffrey Wooldridge, Introductory Econometrics: A Modern Approach, (Thompson 4th ed. 2009)

Jerry Hausman & Greg Leonard, *Efficiencies from the Consumer Viewpoint,* 17(3) George Mason Law Review 707-727 (1999)

John Harsanyi, *On the Rationality Postulates underlying the Theory of Cooperative Games,* 5(2) Journal of Conflict Resolution 179–196 (1961)

John M. Newman, *Antitrust in Zero-Priced Markets: Foundations*, 164 Penn Law Review 149-206 (2015)

John Newman, *The Output-Welfare Fallacy: A Modern Antitrust Paradox*, 107(2) Iowa L. Rev. (2022 forthcoming)

John Peters, Jimikaye Beck, Jan Lande, Zhaoxing Pan, Michelle Cardel, Keith Ayoob, and James O. Hill, *Using Healthy Defaults in Walt Disney World Restaurants to Improve Nutritional Choices*, 1(1) Journal of the Association for Consumer Research 92-103 (2016)

Jonas Bjornerstedt & Frank Verboven, *Merger simulation with nested logit demand*, 14(3) STATA Journal 511-540 (2014)

Jonathan B. Baker & Fiona Scott Morton, *Antitrust Enforcement Against Platform MFNs*, 127(7) Yale Law Journal 2176-2202, 2177 (2017)

Joseph Farrell & Paul Klemperer, *Coordination and Lock-In: Competition with Switching Costs and Network Effects* 3 Handbook of Industrial Organization 1970-2056, 1970 (2007)

Kevin Caves & Hal Singer, *Competing Approaches to Antitrust: An Application in the Payment Card Industry*, 27(3) George Mason Law Review 823-861 (2020)

Kevin Caves and Hal Singer, *On the Utility of Surrogates for Rule of Reason Cases*, CPI Antitrust Chronicle, (2015)

Kim et al., *Two-sided platform competition with multihoming agents: An empirical study on the daily deals market*, 41 Information Econ. and Policy 36-53 (2017)

Louis Kaplow, *Why (Ever) Define Markets?,* 124 Harvard Law Review 437 (2010)

Marc Rysman, *The Economics of Two-Sided Markets*, 23(3) Journal of Economic Perspectives 125-143, 126 (2009)

Mark Armstrong & Julian Wright, *Two-sided markets, competitive bottlenecks and exclusive contracts*, 32 Economic Theory 353-380, 359 (2007)

Marvin B. Lieberman & David B. Montgomery, *Conundra and progress: Research on entry order and performance*, 46 Long Range Planning 312–324 (2013), *available at* www.sciencedirect.com/science/article/abs/pii/S0024630113000344?via%3Dihub

Michael Katz & William Rogerson, *The Applications Barrier to Entry and Its Implication for the Microsoft Remedies: Comment on Iansiti and Richards*, 75 Antitrust Law Rev. 723-738 (2008-09)

Michael Katz and Carl Shapiro. *Critical loss: Let's tell the whole story*, Antitrust 17 (2002) 49

Michael Katz and Harvey Rosen, Microeconomics 3rd ed. 329-330 (Irwin/McGraw-Hill 1998)

N. Gregory Mankiw, Principles of Microeconomics (Cengage Learning 8th ed. 2018)

Nathan Miller, Marc Remer, & Gloria Sheu, *Using cost pass-through to calibrate demand*, 118 Economics Letters 451-454 (2013)

Patrick Greenlee, David Reitman, and David S. Sibley, *An antitrust analysis of bundled loyalty discounts*, 26 International Journal of Industrial Organization 1132-1152 (2007)

Paul Klemperer, *Markets with Consumer Switching Costs*, 102(2) Quarterly Journal of Economics 375-394 (1987)

Richard G. Price, *Market Power and Monopoly Power in Antitrust Analysis*, 75(1) Cornell Law Review 190-217 (1989)

Risto Gogoski, *Payment systems in economy - present end future tendencies*, 44 Procedia - Social and Behavioral Sciences 436–445 (2012)

Robert Bork, The Antitrust Paradox: A Policy At War with Itself (Bork Publishing 2021)

Rong Ding & Julian Wright, *Payment Card Interchange Fees and Price Discrimination*, 65 J. Ind. Org. 39 (2017)

Rong Ding, *Merchant Internalization Revisited,* 125 Econ. Letters 347 (2014)

Sheng-Ping Yang, *Identifying a dominant firm's market power among sellers of a homogeneous product: an application to Alcoa*, 34 Applied Economics 1411-1419 (2002)

Simran Kahai, David Kaserman & John Mayo, *Is the "Dominant Firm" Dominant? An Empirical Analysis of AT&T's Market Power*, 39 Journal of Law & Economics 499-517 (1996)

Steven Berry, *Estimating Discrete Choice Models of Product Differentiation* 25(2) RAND Journal of Economics 242–262 (1994)

Susan Athey & Fiona Scott Morton, *Platform Annexation*, Stanford Inst. for Econ. Policy Research Working Paper 21-015 (March 2021)

Theon van Dijk & Frank Verboven, *Quantification of Damages*, 3 Issues In Competition Law And Policy 2331, 2342 (ABA Section of Antitrust Law 2008)

Thomas A. Burnham, Judy K. Frels, and Vijay Mahajan, *Consumer Switching Costs: A Typology, Antecedents, and Consequences*, 31(2) Journal of the Academy of Marketing Science 109-126 (2003)

Thomas Eisenmann, Geoffrey Parker, and Marshall W. Van Alstyne, *Strategies for Two-Sided Markets*, Harvard Business Review 3-4 (2006)

William Landes & Richard Posner, *Market Power in Antitrust Cases,* 94(5) Harvard Law Review 937-996 (1981)

Zachary Brown, Nick Johnstone, Ivan Haščič, Laura Vong, and Francis Barascud, *Testing the effect of defaults on the thermostat settings of OECD employees*, 39 Energy Economics 128-134 (2013)

PUBLICLY AVAILABLE MATERIALS

2004 Founder's IPO Letter, From the S-1 Registration Statement, *available at* abc.xyz/investor/founders-letters/2004-ipo-letter/

Abhin Mahipal, *Survey: 18% of Android users would consider switching to iPhone 13, but this is down 15% from last year*, Sellcell, (Aug. 31, 2021), *available at* www.sellcell.com/blog/survey-18-percent-of-android-users-would-consider-switching-to-iphone-13/

Abigail Ng, *Smartphone users are waiting longer before upgrading — here's why*, CNBC, (May 17, 2019), *available at* www.cnbc.com/2019/05/17/smartphone-users-are-waiting-longer-before-upgrading-heres-why.html

Adyen (www.adyen.com/about)

Adyen*, Interchange fees explained, available at www.adyen.com/blog/interchange-fees-explained*

Adyen, *Pricing, available at* www.adyen.com/pricing; www.adyen.com/customers

Akhilesh Ganti, *Chicago School of Economics*, Investopedia (Feb. 21, 2021), *available at* www.investopedia.com/terms/c/chicago_school.asp

Alan Santillan, *Mobile Software: Why It's Only Android vs. iOS in 2020*, Learn Hub, (July 30, 2018), *available at* learn.g2.com/android-vs-ios

Alex Cranz, *There are over 3 billion active Android devices. That's a lot of smartphones*, The Verge, (May 18, 2021), *available at* www.theverge.com/2021/5/18/22440813/android-devices-active-number-smartphones-google-2021

Alex Hern, *Microsoft to let developers keep all their Windows app store revenue*, The Guardian (June 25, 2021), *available at* www.theguardian.com/technology/2021/jun/25/microsoft-let-developers-keep-all-windows-app-store-revenue#:~:text=Currently%2C%20developers%20who%20sell%20apps,the%20revenue%20with%20the%20company

Allison Kidd, *Technology experiences: what makes them compelling?*, HPLabs Technical Report (2001)

Amanda Swan & Frank Kehl, *How Does Stripe Work? Everything You Need To Know About Processing Payments With Stripe*, Merchant Maverick (July 22, 2021), *available at* www.merchantmaverick.com/how-does-stripe-work/

Amanda Swan, *Which online stores accept PayPal?*, finder (Dec. 28, 2020), *available at* www.finder.com/paypal-store-list

Amazon Pay Sign up, *available at* pay.amazon.com/

Amazon Pay, *Innovative Merchant Payment Services for Small to Medium Businesses, available at* pay.amazon.com/business/small-business

Amazon, Help & Customer Service – Tax on Amazon Prime, *available at* www.amazon.com/gp/help/customer/display.html?nodeId=202036230

Amit Chowdhry, *Average iPhone Price Increases To $687 and Android Decreases To $254, Says Report*, Forbes, (Feb. 3, 2015), *available at* www.forbes.com/sites/amitchowdhry/2015/02/03/average-iphone-price-increases-to-687-and-android-decreases-to-254-says-report/?sh=d9bcd17539e4

Android, *The best of Google, right on your devices, available at* www.android.com/gms/

Angel Au-Yeung, *App Providers Are 'All Afraid' Of Apple's And Google's Market Power, Match Group And Spotify Tell Senate*, Forbes (Apr. 21, 2021), *available at* www.forbes.com/sites/angelauyeung/2021/04/21/app-providers-are-all-afraid-of-apples-and-googles-market-power-match-group-and-spotify-tell-senate/?sh=4a622ae9596c

Apple, *Move from Android to iPhone, iPad or iPod touch, available at* support.apple.com/en-gb/HT201196

Apple, *What is Family Sharing?, available at* support.apple.com/en-us/HT201060

Appointment Booking & Online Scheduling Software - Pricing – BookedIn, *available at* support.bookedin.com/hc/en-us/articles/360028446492-Why-are-the-prices-different-in-App-Store-Google-Play

Aptoide, *About Us, available at* en.aptoide.com/company/about-us

Aptoide*, For Developers, available at* en.aptoide.com/company/developers

Aptoide, *How to download and install Aptoide?, available at* en.aptoide.com/company/faq/how-to-download-install-aptoide

authorize.net, *Pricing, available at* www.authorize.net/sign-up/pricing.html

Bankrate, *Guide to Google Pay* (Jan. 14, 2021), *available at* www.bankrate.com/finance/credit-cards/android-pay-google-pay-guide/#:~:text=While%20there%20is%20no%20charge,you%20use%20a%20credit%20card

Blake Stimac, *How to sideload an app onto your Android phone or tablet*, Greenbot, (July 17, 2014), *available at* www.greenbot.com/article/2452614/how-to-sideload-an-app-onto-your-android-phone-or-tablet.html

Bookedin, *Why are the prices different in App Store/Google Play?* (accessed June 15, 2021), *available at* support.bookedin.com/hc/en-us/articles/360028446492-Why-are-the-prices-different-in-App-Store-Google-Play-

Braintree, *Boost Revenue with a Global Payments Partner, available at* www.braintreepayments.com/

Braintree, *Pricing, available at* www.braintreepayments.com/braintree-pricing

Brittany Vincent, *Valve Introduces New Revenue Split Changes For Steam Sales*, Variety (Dec. 3, 2018), *available at* variety.com/2018/gaming/news/valve-revenue-split-changes-1203078700/

Bryan Falla, *Google Shopping Actions Commission Rates*, GoDataFeed (Oct. 22, 2019), *available at* www.godatafeed.com/blog/google-shopping-actions-commission-structure

C. Scott Brown, *Google Play Points rewards program is real and in Japan right now*, Android Authority (Sept. 18. 2018), *available at* www.androidauthority.com/google-play-points-rewards-905387/

Celena Chong, *Spotify shows its iPhone users how to save $3 by avoiding Apple's App Store*, Business Insider (July 8, 2015), *available at* www.businessinsider.com/spotify-shows-users-how-to-save-3-by-avoiding-apple-app-store-2015-7

Chaim Gartenberg, *Google will reduce Play Store cut to 15 percent for a developer's first $1M in annual revenue*, The Verge, (Mar. 16, 2021), *available at* www.theverge.com/2021/3/16/22333777/google-play-store-fee-reduction-developers-1-million-dollars

Chuck Jones, *Apple's iOS Loyalty Rate Is Lower Than Google's Android, But Apple May Steal More Users Each Year*, Forbes, (Mar. 10, 2018), *available at* www.forbes.com/sites/chuckjones/2018/03/10/apples-ios-loyalty-rate-is-lower-than-googles-android-but-apple-may-steal-more-users-each-year/?sh=29b39ae68a8e

Coda Payments partners Riot Games for payments services across Southeast Asia, The Paypers, (May 4, 2020), *available at* thepaypers.com/ecommerce/coda-payments-partners-riot-games-for-payments-services-across-southeast-asia--1242106

Consumer Intelligence Research Partners, *How Long Do Android Users Own an Android Phone*, (Sep. 21, 2016), *available at* files.constantcontact.com/150f9af2201/a238f4a1-5b70-4853-b21e-226c94104d30.pdf

Consumer Intelligence Research Partners, *Mobile Operating System Loyalty: High and Steady*, (Mar. 8, 2018), *available at* files.constantcontact.com/150f9af2201/4bca9a19-a8b0-46bd-95bd-85740ff3fb5d.pdf

D. Melanson, *Google makes Chrome Web Store available worldwide, adds in-app purchases and flat five percent fee*, Engadget (May 11, 2011), *available at* www.engadget.com/2011-05-11-google-makes-chome-web-store-available-worldwide-adds-in-app-pu.html

D. Tighe, *Distribution of e-commerce payment methods in the United States in 2020*, Statista, *available at* www.statista.com/statistics/935676/payment-methods-used-for-online-transactions-usa/

Dallas Thomas, *Get Easy Updates on Sideloaded Android Apps,* Gadget Hacks, (Dec. 27, 2016), *available at* android.gadgethacks.com/how-to/get-easy-updates-sideloaded-android-apps-0174291/

Daria Dubrova, *9 Alternative Android App Stores*, The App Solutions, *available at* theappsolutions.com/blog/marketing/alternative-android-app-stores/

David Curry, *App Data Report,* Business of Apps (2022)

David Curry, *App Store Data (2022)*, Business of Apps, (Jan. 11, 2022), *available at* www.businessofapps.com/data/app-stores/

David Nield, *Make your phone and computer team up to get more done*, Popular Science, (Oct. 20, 2018), *available at* www.popsci.com/phone-computer-work-together/

Derek Strickland, *Apple's 30% App Store commission is 'supracompetitive,' court declares*, TweakTown (Sept. 11, 2021), *available at* www.tweaktown.com/news/81567/apples-30-app-store-commission-is-supracompetitive-court-declares/index.html

Development Options and Costs, *Business Information Systems: Design an App for That*, Table 7.1 "Various Fixed Costs", *available at* saylordotorg.github.io/text_business-information-systems-design-an-app-for-that/s11-01-development-options-and-costs.html

Douglas Karr, *PayPal Market Share Statistics And Its History of Dominating Online Payment Processing*, MarTech Zone (Aug. 3, 2020), *available at* martech.zone/paypal-statistics-online-payments/

Drew Strojny, *Stripe vs PayPal: Who should you choose?*, Memberful (Jan. 10, 2016), *available at* memberful.com/blog/stripe-vs-paypal/

eBay, *Understanding selling fees, available at* pages.ebay.com/seller-center/get-started/seller-fees.html

Epic Press Release, *The Epic Games store is now live,* (Dec. 6, 2018), *available at* www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live

ET Telecom.com, *South Korea's app market ONE store grows amid Google's Play store policy row* (Feb. 21, 2021), *available at* telecom.economictimes.indiatimes.com/news/south-koreas-app-market-one-store-grows-amid-googles-play-store-policy-row/81135498

Etsy, *Sell, available at* www.etsy.com/sell

European Commission, *Antitrust: Commission sends Statement of Objections to Google on Android operating system and applications* – Factsheet, (Apr. 20, 2016), *available at* ec.europa.eu/commission/presscorner/detail/en/MEMO_16_1484

Featured Customers – Autorize.net, *available at* www.featuredcustomers.com/vendor/authorizenet/customers

giffgaff, *How to set up family sharing on iPhone and Android*, (July 26, 2019), *available at* www.giffgaff.com/blog/how-to-set-up-family-sharing-on-iphone-amp-android/

Giorgio Sardo, General Manager – Microsoft Store, *Building a new, open Microsoft Store on Windows 11,* Microsoft Windows Blogs (Jun. 24, 2021), *available at* blogs.windows.com/windowsexperience/2021/06/24/building-a-new-open-microsoft-store-on-windows-11/

Google - Android Developers Blog, *Listening to Developer Feedback to Improve Google Play* (Sept. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html

Google - Play Console Help, *Supported locations for distributions to Google Play users, available at* support.google.com/googleplay/android-developer/answer/10532353?visit_id=637777015722462270-3131223409&rd=1

Google - Play Console Help, *Understanding Google Play's Payments policy, available at* support.google.com/googleplay/android-developer/answer/10281818?hl=en#zippy=%2Ccan-i-distribute-my-app-via-other-android-app-stores-or-through-my-website%2Ccan-i-communicate-with-my-users-about-alternative-ways-to-pay%2Ccan-i-communicate-with-my-users-about-promotions-on-other-platforms

Google Ads Help, *About App campaigns, available at* support.google.com/google-ads/answer/6247380?hl=en.

Google Developer Program Policy (effective Dec. 1, 2021), *available at* support.google.com/googleplay/android-developer/answer/11498144?hl=en&visit_id=637814760589469507-2803788482&rd=1

Google Developer Program Policy (effective March 1, 2021), *available at* support.google.com/googleplay/android-developer/answer/9934569?hl=en&ref_topic=9877065#zippy=%2Cmarch

Google Merchant Center Help, *New 0% commission fee for selling on Google through Shopping Actions in the US* (July 23, 2020), *available at* support.google.com/merchants/answer/9977875?hl=en

Google Pay, *For Business, available at* pay.google.com/about/business/partners/

Google Play Billing, *available at* developer.android.com/distribute/play-billing

Google Play Console, *Play Media Experience Program, available a*t play.google.com/console/about/mediaprogram/

Google Play Developer Distribution Agreement (as of Jan. 4, 2014)

Google Play Developer Distribution Agreement (as of Sep. 25, 2014)

Google Play Developer Distribution Agreement (effective as of June 12, 2020)

Google Play services, *available at* developer.android.com/distribute/play-services

Google Play, *Google Play has something for everyone, available at* play.google.com/about/howplayworks/?section=about-google-play&content=service-fee

Google Privacy & Terms, *Google Privacy Policy, available at* policies.google.com/privacy?hl=en-US

Google Safety Center, *Ads That Respect Your Privacy, available at* safety.google/privacy/ads-and-data/

Google SEC Form 10-K, for the fiscal year ended Dec. 31, 2007, *available at* www.sec.gov/Archives/edgar/data/1288776/000119312508032690/d10k.htm

Google SEC Form 10-K, for the fiscal year ended Dec. 31, 2020, *available at* www.sec.gov/Archives/edgar/data/1652044/000165204421000010/goog-20201231.htm

Google, *Tax rates and value-added tax (VAT) - Play Console Help, available at* support.google.com/googleplay/android-developer/answer/138000?hl=en

Hulu, *Why was I charged tax?, available at* help.hulu.com/s/article/charged-sales-tax#:~:text=Why%20was%20I%20charged%20tax%3F&text=In%20certain%20jurisdictions%2C%20Hulu%20is,that%20is%20assessing%20the%20tax

Hung Truong, *Compare 9 Online Marketplace Fees* (Sept. 18, 2018), *available at* sellerzen.com/compare-9-online-marketplace-fees

IGI Global, *What is Mobile Devices, available at* www.igi-global.com/dictionary/use-of-apps-and-devices-for-fostering-mobile-learning-of-literacy-practices/18837

Investigation of Competition in Digital Markets: Majority Staff Report and Recommendations, H. Subcomm. on Antitrust, Commercial and Administrative Law of the Comm. on the Judiciary, (Oct. 6, 2020)

J.P. Morgan, *2020 E-commerce Payments Trends Report: US, available at* www.jpmorgan.com/merchant-services/insights/reports/united-states-2020

J.P. Morgan, *E-commerce Payments Trends: United States (2019), available at* www.jpmorgan.com/merchant-services/insights/reports/united-states

Jack Nicas, *How Apple's 30% App Store Cut Became a Boon and a Headache*, New York Times (Aug. 14, 2020), *available at* www.nytimes.com/2020/08/14/technology/apple-app-store-epic-games-fortnite.html

Jack Wallen, *Why is Android more popular globally, while iOS rules the US?*, Tech Republic, (May 12, 2021), *available at* www.techrepublic.com/article/why-is-android-more-popular-globally-while-ios-rules-the-us/

Jason Turnquist, *How Much Does App Development Cost?*, Fyresite, (July 31, 2020), *available at* www.fyresite.com/how-much-does-app-development-cost/

Jennifer Dunn, *Sales Tax by State: Should You Charge Sales Tax on Digital Products?* (Feb. 13, 2018), *available at* www.taxjar.com/blog/sales-tax-digital-products

Jerry Hildenbrand, *Sideloading and Unknown Sources on Android: How to do it and fix errors*, Android Central, (Apr. 16, 2020), *available at* www.androidcentral.com/unknown-sources

Jimmy Westenberg, *How to install the Amazon Appstore on your Android [Android 201]*, AndroidGuys, (Apr. 5, 2014), *available at* www.androidguys.com/tips-tools/install-amazon-app-store-android/

Joel Snyder, *What are the risks of sideloaded Android applications?*, Samsung Knox, (Apr. 20, 2020), *available at* www.samsungknox.com/en/blog/what-are-the-risks-of-sideloaded-android-applications

Jonas DeMuro & Brian Turner, *Best payment gateways of 2021*, Tech Radar (Apr. 20, 2021), *available at* www.techradar.com/best/best-payment-gateways

Jonathan Jaehnig, *What Are Google Play Points and How Can You Use Them?*, Make Use Of, (Apr. 8, 2021), *available at* www.makeuseof.com/what-are-google-play-points/

Judopay (www.judopay.com/about)

Julija A., *US Credit Card Market Share: Facts and Statistics*, Fortunly (Nov. 23, 2021), *available at* fortunly.com/articles/credit-card-market-share/

Kif Leswig, *Bumble, Duolingo lead rally in shares of app developers after Google slashes subscription fees*, CNBC (Oct. 21, 2021), *available at* www.cnbc.com/2021/10/21/google-cuts-app-store-fees-lifting-shares-of-app-developers.html

Kif Leswing, *Apple will cut App Store commissions by half to 15% for small app makers*, CNBC, (Nov. 18, 2020), *available at* www.cnbc.com/2020/11/18/apple-will-cut-app-store-fees-by-half-to-15percent-for-small-developers.html

Kif Leswing, *Google to enforce 30% take from in-app purchases next year*, CNBC (Sept. 28, 2020), *available at* www.cnbc.com/2020/09/28/google-to-enforce-30percent-cut-on-in-app-purchases-next-year.html

Kim Byung-wook, *Google's app billing plan continues to backfire*, The Korea Herald (June 28, 2021), *available at* www.koreaherald.com/view.php?ud=20210628000824

Kim Eun-jung, *Korean app market ONE store eyes global alliance to compete with Google* (Dec. 1, 2019), *available at* en.yna.co.kr/view/AEN20191128004700320

Kim Jung-Min & Chea Sarah, *One Store Gains Ground in Local Android App Market*, Korea JoonAng Daily (Dec. 2, 2020), *available at* koreajoongangdaily.joins.com/2020/12/02/business/industry/One-Store-app-market-Google/20201202175300439.html

Kyle Orland, *Humble Bundle creator brings antitrust lawsuit against Valve over Steam*, Ars Technica (Apr. 30, 2021), *available at* arstechnica.com/gaming/2021/04/humble-bundle-creator-brings-antitrust-lawsuit-against-valve-over-steam/

Lewis Krauskopf, *Swiping their way higher: Visa, Mastercard could be the next $1 trillion companies*, Reuters (January 31, 2020), *available at* www.reuters.com/article/us-visa-mastercard-stocks/swiping-their-way-higher-visa-mastercard-could-be-the-next-1-trillion-companies-idUSKBN1ZU0JA

Lim Young-sin & Choi Mira, *Korea's home-grown integrated app market One Store on global outreach*, Pulse (Nov. 13, 2019), *available at* pulsenews.co.kr/view.php?year=2019&no=938924

Lionel Valdellon, *What Are the Different Types of Mobile Apps? And How Do You Choose?*, CleverTap (Nov. 1, 2020), *available at* clevertap.com/blog/types-of-mobile-apps/

Lucas Mearian, *iOS vs. Android: When it comes to brand loyalty, Android wins*, Computerworld, (Mar. 9, 2018), *available at* www.computerworld.com/article/3262051/ios-vs-android-when-it-comes-to-brand-loyalty-android-wins.html

MacRumors, *Apple Pay, available at* www.macrumors.com/roundup/apple-pay/#:~:text=Some%20of%20Apple's%20partners%20include,Bell%2C%20and%207%2D11

Mansoor Iqbal, App Revenue Data (2022), *Business of Apps*, (Feb. 16, 2022), *available at* www.businessofapps.com/data/app-revenues/

Mark Jansen, Christian de Looper, and Paula Beaton, *PayPal vs. Google Pay vs. Venmo vs. Cash App vs. Apple Pay Cash*, Digital Trends (July 5, 2021), *available at* www.digitaltrends.com/mobile/paypal-vs-google-wallet-vs-venmo-vs-square-cash/

Mark Looi, *On "The Platform Delusion" by Jonathan Knee*, Medium (Dec. 21, 2021), *available at* marklooi.medium.com/on-the-platform-delusion-by-jonathan-knee-a787a672b932

Max Willens, *Cheat sheet: Twitter's acquisition of Revue heats up the battle of the inbox*, Digiday (Jan. 27, 2021), *available at* digiday.com/media/cheat-sheet-twitters-acquisition-of-revue-heats-up-the-battle-of-the-inbox/

Megan Graham & Jennifer Elias, *How Google's $150 billion advertising business works*, CNBC, (May 18, 2021), *available at* www.cnbc.com/2021/05/18/how-does-google-make-money-advertising-business-breakdown-.html

Melanie Hicken, *Are you paying the iTunes tax?*, CNN Money (June 5, 2013), *available at* money.cnn.com/2013/06/05/pf/taxes/itunes-tax/

Microsoft Store, *App Developer Agreement Version 8.7* (Effective July 28, 2021), *available at* query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4OG2b

Mike Peterson, *Amazon following Apple & Google's lead, cutting app store commissions*, AppleInsider (June 17, 2021), *available at* appleinsider.com/articles/21/06/17/amazon-following-apple-googles-lead-cutting-app-store-commissions

Nabila Amarsy, *Switching Costs: 6 Ways to Lock Customers Into Your Ecosystem*, Strategyzer, (July 27, 2015), *available at* www.strategyzer.com/blog/posts/2015/7/27/switching-costs-6-strategies-to-lock-customers-in-your-ecosystem

Neal St. Anthony, *Vanco Payment Solutions grows as electronic bridge between churches, charities and their donors*, Star Tribune (Aug. 31, 2020), *available at* www.startribune.com/vanco-payment-solutions-grows-as-electronic-bridge-between-churches-charities-and-their-donors/572250712/

Netflix, *Taxes on your Netflix membership, available at* help.netflix.com/en/node/50068#:~:text=The%20Netflix%20advertised%20price%20does,membership%20includes%20streaming%20and%20games

Nick Statt, *Apple Just Kicked Fortnite Off the App Store*, The Verge (Aug. 13, 2020), *available at* www.theverge.com/2020/8/13/21366438/apple-fortnite-ios-app-store-violations-epic-payments

Nick Statt, *Tinder is now bypassing the Play Store on Android to avoid Google's 30 percent cut*, The Verge (Jul. 19, 2019), *available at* www.theverge.com/2019/7/19/20701256/tinder-google-play-store-android-bypass-30-percent-cut-avoid-self-install

OECD Policy Roundtable, *Market Definition 2012*, Organisation for Economic Co-operation and Development (October 11, 2012), *available at* www.oecd.org/daf/competition/Marketdefinition2012.pdf

Oxera, *Paying up: the new economics of payment systems* (Jun. 30, 2020), *available at* www.oxera.com/insights/agenda/articles/paying-up-the-new-economics-of-payment-systems/

Patrick Icasas, *Your Customer is a Churn Risk (And Here's How We Know)*, Catalyst (July 12, 2021), *available at* catalyst.io/blog/your-customer-is-a-churn-risk-and-heres-how-we-know

Paysafe (www.paysafe.com/us-en/)

Pixel Phone Help, *Transfer Data from an iPhone to a Pixel, available at* support.google.com/pixelphone/answer/7129740#what_doesnt_copy&zippy=%2Cwhat-wont-copy-during-setup

Play Console Help, *How to use Play Console, available at* support.google.com/googleplay/android-developer/answer/6112435?hl=en#zippy=%2Cstep-pay-registration-fee

Play Console Help, *Service fees, available at* support.google.com/googleplay/android-developer/answer/112622?hl=en

Poshmark, *What are the fees for selling on Poshmark, available at* support.poshmark.com/s/article/297755057?language=en_US

POSQuote.com, *Fattmerchant vs. Square: Which Payment Processor Is Best?*, *available at* posquote.com/review/fattmerchant-vs-square

ProtonMail wades into U.S. antitrust war, Axios, July 22, 2021

Rachel Rickard Straus, *Apple faces landmark legal claim that could pay out to millions: Rip-off that adds 30% to price of smartphone apps*, This is Money, (Dec. 11 2021), *available at* www.thisismoney.co.uk/money/bills/article-10299235/Rip-adds-30-price-smartphone-apps.html

Revenue Share, *Catappult App Distribution Console, available at* docs.catappult.io/docs/distribution-and-revenue-share

Rob Enderle, *How Microsoft failed with Windows 10 Mobile*, Computerworld, (Jan. 24, 2019), *available at* www.computerworld.com/article/3336057/how-microsoft-failed-with-windows-10-mobile.html

Ron Amadeo, *Google announces crackdown on in-app billing, aimed at Netflix and Spotify*, Ars Technica (Sept. 20, 2020)

Sameer Samat - Vice President, Product Management, *Evolving our business model to address developer needs*, Android Developers Blog (Oct. 21, 2021), *available at* android-developers.googleblog.com/2021/10/evolving-business-model.html

Sameer Samat – Vice President, Product Management, *Boosting Developer Success on Google Play* (Mar. 16, 2021), *available at* android-developers.googleblog.com/2021/03/boosting-dev-success.html

Sameer Samat, *Listening to Developer Feedback to Improve Google Play*, Android Developers Blog, (Sep. 28, 2020), *available at* android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html

Sarah Perez, *Amazon's Appstore lowers its cut of developer revenue for small businesses*, *adds AWS credits,* TechCrunch (June 17, 2021), *available at* techcrunch.com/2021/06/17/amazons-appstore-lowers-its-cut-of-developer-revenue-for-small-businesses-adds-aws-credits/

Sensor Tower, *2019 – 2023 Mobile Market Forecast,* (2019)

Sensor Tower, *2021 – 2025 Mobile Market Forecast*, (2021), *available at* go.sensortower.com/rs/351-RWH-315/images/Sensor-Tower-2021-2025-Market-Forecast.pdf

Sensor Tower, *Global Consumer Spending in Mobile Apps Reached $133 Billion in 2021, Up Nearly 20% from 2020*, (Dec. 2021), *available at* sensortower.com/blog/app-revenue-and-downloads-2021

Shahar Ziv, *Here's Why Your Apple App Store Purchases May Be A Ripoff*, Forbes, July 8, 2020, *available at* www.forbes.com/sites/shaharziv/2020/07/08/heres-why-your-apple-app-store-purchases-may-be-a-ripoff/?sh=77c6e9872007

Shannon Vissers, *How Much Does Clover Cost? Your Ultimate Guide To Clover Fees*, Merchant Maverick (Aug. 19, 2021), *available* at www.merchantmaverick.com/clover-pos-cost/

Sherisse Pham*, Google now has two apps in China, but search remains off limits,* CNN Business, (May 31, 2018)*, available at money.cnn.com/2018/05/31/technology/google-in-china-files-app/index.html*

Spotify, *Does the price for Premium include tax?, available at* support.spotify.com/us/article/sales-tax/

Square, *Payment Platforms, available at* squareup.com/us/en/payments/payment-platform

Square, *Payments, available at* squareup.com/us/en/payments/pricing

Square, *Your all-in-one restaurant POS system, available at* squareup.com/us/en/point-of-sale/restaurants

Statcounter, *Mobile Operating System Market Share United States of America*, (accessed Feb. 4, 2022), *available at* gs.statcounter.com/os-market-share/mobile/united-states-of-america

Statcounter, *Mobile Operating System Market Share Worldwide*, (accessed Feb. 2022), available at gs.statcounter.com/os-market-share/mobile/worldwide/#yearly-2019-2019-bar

Statement by Executive Vice-President Margrethe Vestager on the Statement of Objections sent to Apple on App Store rules for music streaming providers (Apr. 30, 2021)

Statista, *Advertising revenue of Google from 2001 to 2020, available at* www.statista.com/statistics/266249/advertising-revenue-of-google/

Statista, *Combined global Apple App Store and Google Play Store app downloads from 1st quarter 2015 to 4th quarter 2021, available at* www.statista.com/statistics/604343/number-of-apple-app-store-and-google-play-app-downloads-worldwide/

Statista, *Market share of mobile operating systems in China from January 2013 to December 2021**, *available at* www.statista.com/statistics/262176/market-share-held-by-mobile-operating-systems-in-china/

Statista, *Mobile operating systems' market share worldwide from January 2012 to June 2021*, *available at* www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/

Statista, *Number of apps available in leading app stores as of 2021, available at* www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

Statista, *Number of mobile apps downloaded worldwide from 2016 to 2020, available at* www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/

Stax Payments – Healthcare, *available at* staxpayments.com/healthcare/

Stephanie Chevalier*, Which form of payment do you use most often for online shopping?,* Statista*, available at www.statista.com/statistics/448712/online-shopping-payment-method-preference-usa/*

Substack, *Going Paid, available at* substack.com/going-paid

Terry Stancheva, *17 App Revenue Statistics – Mobile Is Changing the Game in 2021*, TechJury, (Feb. 6, 2022), *available at* techjury.net/blog/app-revenue-statistics/#gref

Testimony for the S. Judiciary Committee, Subcomm. on Competition Policy, Antitrust, and Consumer Rights (Horacio Gutierrez, Head of Global Affairs & Chief Legal Officer, Spotify, Apr. 21, 2021)

Testimony for the S. Judiciary Committee, Subcomm. on Competition Policy, Antitrust, and Consumer Rights (Jared Sine, Chief Legal Officer, Match Group, Apr. 21, 2021)

The new rules of the 'creator economy', Economist (May 8, 2021), *available at* www.economist.com/briefing/2021/05/08/the-new-rules-of-the-creator-economy

Tom McKay, *Twitter Wants to Be Substack Now*, Gizmodo (Jan. 26, 2021), *available at* gizmodo.com/twitter-wants-to-be-substack-now-1846136057

Tom Warren, *Microsoft shakes up PC gaming by reducing Windows store cut to just 12 percent*, The Verge, (Apr. 29, 2021), *available at* www.theverge.com/2021/4/29/22409285/microsoft-store-cut-windows-pc-games-12-percent

Travers Korch, *How tech ecosystems lock you into costs*, Bankrate, (Sep. 30, 2014), *available at* web.archive.org/web/20210602055404/https:/www.bankrate.com/finance/smart-spending/tech-ecosystems-cost-you-1.aspx%20

Van Vicente, *What Does Cross-Platform Mean for Gaming and Other Apps?*, How-To Geek, (Oct. 9, 2021), *available at* www.howtogeek.com/752370/what-does-cross-platform-mean-for-gaming-and-other-apps/

Vanco, *Pricing, available at* www.vancopayments.com/egiving/pricing

Vangie Beal, *What Are Examples of Mobile Operating Systems?*, Webopedia (Jan. 27, 2022), *available at* www.webopedia.com/insights/mobile-os-and-different-types/

Verizon Communications, *Verizon Business offers touchless payment capability with Clover from Fiserv*, GlobeNewswire (Dec. 10, 2020), *available at* www.globenewswire.com/news-release/2020/12/10/2143226/0/en/Verizon-Business-offers-touchless-payment-capability-with-Clover-from-Fiserv.html/

VPNDada, *How to Access Google Play Store in China, available at* www.vpndada.com/access-google-play-store-china/

Walmart Marketplace, *Referral Fees, available at* marketplace.walmart.com/referral-fees/

WEPC, *Console Gaming Statistics 2022*, (Jan. 20, 2022), *available at* www.wepc.com/statistics/console-gaming/

Yoon Seon-hoon, *Google to abide by the 'forced in-app payment law'*, iNews24 (Nov. 4, 2021), *available at* n.news.naver.com/article/031/0000633296

TRIAL MATERIALS

Defendants' Answers and Objections to Developer Plaintiffs' First Set of Interrogatories to Defendants (July 6, 2021)

[redacted]

APPENDIX 3: EXTENSION OF TWO-SIDED MARKET MODEL

285. In what follows, I derive the expressions used in Parts V.B, V.E, and Appendix 4 for demonstrating impact in the particular two-sided setting relevant to the instant case. This "applied" modeling is an extension of the "foundational" model in Rochet and Tirole 2003. I discuss two cases: (1) a case where competition occurs only with respect to the take rate t, and (2) a case where competition occurs only with respect to the buyer-side price P_B.

286. In Rochet and Tirole 2003, the monopolist platform operator maximizes profit (denoted π^0) defined as:

(A.1)
$$\pi^0 = (P_B + P_S - C)D_B(P_B)D_S(P_S)$$

where P_B and P_S are buyer- and seller-side platform prices, respectively; C is marginal transaction cost, and $D_B(P_B)$ and $D_S(P_S)$ are demand functions for buyers and sellers, respectively.[594] I extend the model (1) to accommodate Google charging a percentage take rate on the developer (seller) side, (2) to allow the average App or In-App Content (product) price (set by developers) to be a function of the take rate, and (3) to allow consumer (buyer) demand to be a function of the net App or In-App Content (product) price, defined as the sum of the product price and the platform price. The new profit function (denoted π) can be written as:

(A.2)
$$\pi = (P_B + tS(t) - C)D_B(S(t) + P_B)D_S(t)$$

where $S(t)$ is the price of paid App downloads and t is the take rate or portion of consumer spend that is retained by Google.

287. Before continuing the exposition, it is useful to define the pass-through parameter γ which I refer to throughout:

(A.3)
$$\gamma = \frac{\Delta SQ}{\Delta vQ} = \frac{\Delta S}{\Delta v}$$

where ΔS is the dollar change in product price to consumers and Δv is the dollar change in costs (including commissions paid to Google) to developers.

594. Rochet & Tirole at 996.

A. Case 1: Platform Operator Maximizes Profit Only With Respect to the Take Rate (Buyer-Side Platform Price Is Held Fixed)

288. In this subsection, I present the derivations used to arrive at Equations (V.3) and (V.5) in the report. If the platform operator maximizes profit with respect to the take rate, holding the buyer-side platform price fixed, the following first-order condition[595] follows from (A.2):

(A.4)
$$\frac{S + tS'}{P_B + tS - C} - \frac{\varepsilon_{B,t}}{t} - \frac{\varepsilon_{S,t}}{t} = 0$$

where S' is the change in the product price with respect to the take rate, and the "take-rate elasticities" are:

(A.5)
$$\varepsilon_{B,t} = -\frac{tS'D_B'}{D_B}$$

(A.6)
$$\varepsilon_{S,t} = -\frac{tD_S'}{D_S}$$

D_B' and D_S' are first derivatives of the buyer and seller demand functions. Re-arranging (A.4):

$$\frac{S + tS'}{P_B + tS - C} = \frac{\varepsilon_{B,t} + \varepsilon_{S,t}}{t}$$

Inverting each side:

$$\frac{P_B + tS - C}{S + tS'} = \frac{t}{\varepsilon_{B,t} + \varepsilon_{S,t}}$$

Dividing by t gives Equation (V.3):

(A.7), (V.3)
$$\frac{P_B + tS - C}{tS + t^2S'} = \frac{1}{\varepsilon_{B,t} + \varepsilon_{S,t}}$$

Let k be a developer's marginal cost. The per-unit cost to developers from the take rate is equal to $tS(t)$, implying that the change in marginal cost resulting from a change in the take rate is $\frac{\partial k}{\partial t} = S(t) + tS'(t)$. Setting $\Delta v = S(t) + tS'(t)$ and $\Delta S = S'(t)$ (the changes in price and cost, respectively, resulting from a change in the take rate) in expression (A.3) gives $\gamma = \frac{S'(t)}{S(t)+tS'(t)}$. Re-arranging gives the following expression for $S'(t)$:

(A.8)
$$S'(t) = \frac{\gamma}{1 - t\gamma} S(t)$$

595. A first-order condition describes the point at which profit maximization is achieved and is a commonly used tool in economic modeling. *See, e.g.*, JEFFREY M. PERLOFF, MICROECONOMICS A-34 (Pearson 7th ed. 2015). As in Rochet and Tirole, I apply the log-transform to the profit function, then take the derivative with respect to t.

289. In a monopoly setting, elasticities reflect that the platform operator faces no competition. In a competitive setting, profit is maximized with respect to "residual" demand, defined as the demand curve faced by the platform operator in the presence of competition.[596] To determine the competitive equilibrium condition, I replace demand functions $D_B(S(t) + P_B)$ and $D_S(t)$ in expression (A.2) with residual demand functions and repeat steps (A.4) through (A.7). Residual demand functions are formally defined as the difference between market demand and the quantity being supplied by the firm's rivals:

(A.9)
$$RD_B(S(t) + P_B) = D_B(S(t) + P_B) - Q_{R,B}(S(t) + P_B)$$

(A.10)
$$RD_S(t) = D_S(t) - Q_{R,S}(t)$$

where $Q_{R,B}(S(t) + P_B)$ and $Q_{R,S}(t)$ represent the amount of product supplied by the platform's rivals at price $S(t) + P_B$ and take rate t.[597] Steps (A.4) through (A.7) using the residual demand curves then give the analogous competitive expression used in Equation (V.5):

(A.11), (V.5)
$$\frac{P_B + tS - C}{tS + t^2S'} = \frac{1}{\varepsilon_{OB,t} + \varepsilon_{OS,t}}$$

where $\varepsilon_{OB,t}$ and $\varepsilon_{OS,t}$ are "own-brand" elasticities on the buyer (consumer) and seller (developer) sides, respectively, reflecting the change in quantity demanded from consumers for transaction on the firm's (Google's) platform in response to a change in the take rate and in the presence of competition.

290. To solve for the product price in the competitive setting, I use the pass-through equation (A.3), and note that the change in the total commissions (from t^MS^M to t^CS^C) from a change in the take rate satisfies:

(A.12)
$$\gamma = \frac{\Delta S}{\Delta v} = \frac{S^M - S^C}{t^MS^M - t^CS^C}$$

where superscripts ^{M} and ^{C} denote the monopoly and competitive cases, respectively. Re-arranging allows expression of the product price in the competitive world in terms of the competitive take rate and monopoly inputs:

(A.13), (V.8)
$$S^C = S^M \frac{1 - \gamma t^M}{1 - \gamma t^C}$$

596. *See, e.g.*, Landes & Posner at 985.
597. *Id.*

Using this expression, I can then solve for the competitive take rate that satisfies (A.10), having estimated the other inputs (S^M, C, and competitive elasticities).

B. Case 2: Platform Operator Maximizes Profit Only With Respect to the Buyer-Side Platform Price (Take Rate Is Held Fixed)

291. Using the same objective function (A.2) but maximizing profit with respect to the buyer-side platform price yields the following first-order condition, now taken with respect to P_B:

(A.14)
$$\frac{1}{P_B + tS(t) - C} - \frac{\varepsilon_B}{(S + P_B)} = 0$$

where ε_B is the price elasticity of demand for App products (taken with respect to the net price $S + P_B$):

(A.15)
$$\varepsilon_B = -\frac{(S + P_B)D_B{}'}{D_B}$$

Re-arranging (A.13) gives Equation (V.11):

(A.16), (V.11)
$$\frac{P_B + tS - C}{S + P_B} = \frac{1}{\varepsilon_B}$$

292. To solve for the competitive equilibrium condition, I replace demand functions $D_B(S(t) + P_B)$ and $D_S(t)$ with residual demand functions (defined in Equations A.9 and A.10) in expression (A.2) and repeat steps (A.14) through (A.16). This gives the competitive expression used in Equation (V.12):

(A.17), (V.12)
$$\frac{P_B + tS - C}{S + P_B} = \frac{1}{\varepsilon_{OB}}$$

Note that the product price elasticity of demand is related to the *take rate* elasticity of demand (given by Equation (A.5)):

$$\varepsilon_B = -\frac{tS'D_B'}{D_B} * \frac{(S + P_B)}{S'}$$

or

(A.18)
$$\varepsilon_B = \varepsilon_{B,t}\frac{(S + P_B)}{tS'}$$

The analogous expression using own-brand elasticities is:

(A.19)
$$\varepsilon_{OB} = \varepsilon_{OB,t}\frac{(S + P_B)}{tS'}$$

APPENDIX 4: RESULTS OF THE SINGLE MARKET TAKE RATE MODEL

293. In Section V.B I present a model in which the locus of platform competition in the Android App Distribution Market was the take rate. That is, absent Google's anticompetitive restrictions, I model the extent to which Google would lower its take rate in response to competition. In this model, I hold the buyer-side subsidy fixed at its observed proportion to the paid App download price. In Section V.E I presented an alternative model in which Google increases its loyalty points program for consumers to encourage their use of the Play Store and Google Play Billing rather than using a competing source of Apps or In-App Content. In this buyer-side platform competition model, I model competition with respect to a single per-unit consumer subsidy across both initial paid App downloads and purchases of In-App Content, as Google's present (though small in magnitude) loyalty points program uses this structure (rather than having two different points programs). In this model, I hold the take rate fixed at its observed monopoly level.

294. Here I present a third scenario (the "single market take rate model") where competition occurs only with respect to a take rate (holding the buyer-side subsidy fixed) that applies to a single combined market. The single market take rate model follows the same steps as the Android App Distribution Market take rate model presented in Part V.B; however, in the single market take rate model presented here, the platform operator maximizes profit by choosing a take rate that applies to all transactions (both initial downloads and in-App purchases). Because of this distinction, input values differ across models.

295. My sources and methods for obtaining the monopoly scenario inputs shown in Equation (V.3) are:

- P_B^M is equal to the (negative) price charged by Google to consumers for transactions made on its platform in the monopoly scenario. I compute the average value of this price as the sum of all promotions paid by Google for transactions made in both the Android App Distribution Market and In-App Aftermarket, divided by the total quantity of paid Apps downloaded in the Android App Distribution Market and purchases of In-App Content in the In-App Aftermarket, as observed in Google's transaction records.

- t^M is equal to the observed take rate, computed as the sum of all revenue retained by Google in the Android App Distribution Market and In-App Aftermarket divided by the sum of total revenue spent by consumers in the Android App Distribution Market and In-App Aftermarket (prior to Google's promotional expenditures, which are captured by P_B^M).

- S^M is equal to the average price charged for paid Apps in the Android App Distribution Market and In-App Content in the In-App Aftermarket in the monopoly setting, calculated as the total amount of revenue spent by consumers (prior to receiving promotions from Google) in the Android App Distribution Market and In-App Aftermarket divided by the total quantity of paid Apps downloaded in the Android App Distribution Market and purchases of In-App Content in the In-App Aftermarket, as observed in Google's transaction records.

- Marginal cost C represents the incremental cost incurred by Google in executing a transaction. I refer to Google's financial data to infer this value, which suggests that

transaction fees and direct costs that Google records for the Play Store (excluding content costs) are approximately [redacted] percent of consumer expenditures.[598]

- γ is equal to the change in the App price S charged to consumers with respect to a change in developers' costs (including the cost imposed on developers through Google's take rate), also known as the pass-through rate. This parameter is discussed in detail in Section V.D, where I estimate its value at 89.9 percent.

- S'^M represents the change in the product price resulting from a small change in the take rate. I solve for S'^M in terms of the take rate and pass-through rate: $S'^M = \frac{\gamma}{(1-t^M\gamma)} S^M$. Appendix 3 contains a derivation of this expression.

- $\varepsilon^M_{B,t}$ and $\varepsilon^M_{S,t}$ are the take-rate elasticities of demand for transactions in the Android App Distribution Market and In-App Aftermarket from consumers and developers, respectively, in the presence of Google's monopoly. Given the other inputs to the monopoly model, the value of the sum $\varepsilon^M_{B,t} + \varepsilon^M_{S,t}$ is implied by Equation (V.3).

296. I hold inputs C and γ fixed between the monopoly and competitive scenarios. My sources and methods for obtaining the remaining inputs to the competitive scenario expression shown in Equation (V.5) are:

- t^C is equal to the but-for (competitive) take rate. I solve for the but-for take rate by finding the value that satisfies Equation (V.5) given the remaining inputs.

- S^C is the price of paid App downloads and In-Ap Content that developers would charge in a competitive scenario. S^C can be calculated in terms of other inputs (S^M, t^M, t^C, γ) according to Equation (V.8).

- S'^C represents the change in the product price resulting from a small change in the take rate in the competitive setting. I solve for S'^C in terms of the take rate and pass-through rate: $S'^C = \frac{\gamma}{(1-t^C\gamma)} S^C$. Appendix 3 contains a derivation of this expression.

- $\varepsilon^C_{OB,t}$ and $\varepsilon^C_{OS,t}$ are the "own-brand" take-rate elasticities of demand for transactions in the Android App Distribution Market and In-App Aftermarket for consumers and developers, respectively, in the presence of competition. Following the same steps taken in Section V.B.3, I estimate that Google's take rate elasticities shift from a value of 2.11 (in the monopoly setting, as calculated using Equation (V.3)) to 2.49 in the competitive setting. Because this model is applied to the combined Android App Distribution Market and In-App Aftermarket, elasticity values vary slightly from those shown in Table 3 (which reflects the Android App Distribution Market only).

- P^C_B is equal to the (negative) price charged by Google to consumers for transactions made in the Android App Distribution Market and In-App Aftermarket in the competitive

598. *See* Section V.C, GOOG-PLAY-000416245 (includes all Direct COS & Direct OpEx).

scenario. I hold the buyer-side platform price fixed in proportion to the product price: $P_B^C = \left(\frac{P_B^M}{S^M}\right) S^C$.

297. Table A4 shows the results of the single market take rate model. At a pass-through rate of $\gamma = 89.9$ percent, the resulting but-for average price of paid App downloads and In-App purchases is [redacted], down from the observed price of [redacted] (net of Google's promotional expenditures to consumers). This difference results in an average overcharge to consumers of [redacted], and aggregate damages of $2.35 billion (equal to [redacted] times [redacted] paid App and In-App purchase transactions made) as a result of Google's restrictions, across the Class Period (August 16, 2016, through December 31, 2020).

TABLE A4: SINGLE MARKET TAKE RATE MODEL, IMPACT AND DAMAGES
(8/16/2016 – 12/31/2020)

Actual World (Monopoly, Eqn. (V.3))

#	Input	Description	Value	Source/Notes
[1]		Consumer Expenditure (US)	[redacted]	GOOG-PLAY-005535886 (US Consumers)
[2]		Google Revenue (US)		GOOG-PLAY-005535886 (US Consumers)
[3]		Google Promotional Expenditures (US)		GOOG-PLAY-005535886 (US Consumers)
[4]		Android App Distribution (Paid) and In-App Aftermarket Transactions		GOOG-PLAY-005535886 (US Consumers)
[5]=[1]/[4]	S^M	App Product Price		Calculated
[6]=[2]/[1]	t^M	Take Rate		Calculated
[7]=-[3]/[4]	P_B	Buyer-side Platform Price		Calculated
[8]=[5]+[7]	$S^M + P_B$	App Product Price Net of Promotions		Calculated
[9]	C	Marginal Cost		GOOG-PLAY-000416245
[10]	γ	Pass-through rate		Estimated (See Table 8)
[11]	$\varepsilon^M_{B,t} + \varepsilon^M_{S,t}$	Take Rate Elasticities of Demand		Calculated (Eqn. (V.3))

But-For World (Competitive, Eqn. (V.5))

#	Input	Description		Source/Notes
[12]	S^C	App Product Price		Calculated (Eqn.(V.8))
[13]	t^C	Take Rate		Calculated (Eqn.(V.5))
[14]=([7]/[5])* [12]	P_B	Buyer-side Platform Price		Calculated
[15]=[12]+[14]	$S^C + P_B$	App Product Price Net of Promotions		Calculated
[16]=[9]	C	Marginal Cost		GOOG-PLAY-000416245
[17]=[10]	γ	Pass-through Rate		Estimated (See Table 8)
[18]	$\varepsilon^C_{OB,t} + \varepsilon^C_{OS,t}$	Take Rate Elasticities of Demand		Economic theory/ empirical studies
[19]=[8]-[15]		Consumer Savings Per Transaction		Calculated
[20]=[19]*[4]		**Aggregate Damages**		Calculated

APPENDIX 5: STANDARD SSNIP TEST INDICATES IN-APP AFTERMARKET PURCHASES IS A RELEVANT ANTITRUST MARKET

298. In this Appendix, I assess whether the In-App Aftermarket constitutes its own relevant antitrust market. In this section, I use an industry standard market definition exercise to show it is indeed the case that it does

299. Economists and antitrust agencies use a market definition exercise to evaluate "a customer's ability and willingness to substitute away from one product to another in response to a price increase" by a hypothetical monopolist.[599] The DOJ and FTC's *Horizontal Merger Guidelines* define a relevant antitrust product market as a product or group of products that, if controlled by a hypothetical monopolist, could profitably sustain a small but significant and non-transitory increase in price ("SSNIP") over the competitive price level.[600] A SSNIP is usually assessed as a five-percent increase in price over competitive levels.[601]

300. Economists perform what is known as a "critical loss analysis" to assess the profitability of a SSNIP of five percent on a category of goods or services.[602] All else equal, raising a product's price increases its profit margin. But for virtually all products, higher prices also result in fewer sales. How *many* fewer sales a product earns in response to a change in price is a product's "own-price demand elasticity," which is defined as the percentage decrease in quantity demanded that results from a one-percent increase in price.[603] A critical loss analysis determines if a product has sufficient outside competition to deter a hypothetical monopolist from profitably raising prices. If not, the product is its own relevant antitrust market.

301. A critical loss analysis can be broken into three parts. First, an economist calculates the largest theoretical loss in sales (in percentage terms) a hypothetical monopolist could sustain above competitive levels before a five-percent price increase would become unprofitable. The critical loss formula is based on the profit margin of the hypothetical monopolist (in percentage terms) and the price increase being tested. For a five-percent price increase, the critical loss is calculated as:[604]

$$Critical\ Loss = \frac{5\%}{(Margin + 5\%)}$$

---

599. *Merger Guidelines* §4.

600. *Id.* §4.1.1.

601. *Id.* §4.1.2 ("The Agencies most often use a SSNIP of five percent of the price paid by customers for the products or services to which the merging firms contribute value. However, what constitutes a "small but significant" increase in price, commensurate with a significant loss of competition caused by the merger, depends upon the nature of the industry and the merging firms' positions in it, and the Agencies may accordingly use a price increase that is larger or smaller than five percent.").

602. *Merger Guidelines* §4.1.3. *See also* Daniel O'Brien and Abraham Wickelgren. *A critical analysis of critical loss analysis*, 71 ANTITRUST L.J. (2003) 161 [hereafter O'Brien & Wickelgren]; Michael Katz and Carl Shapiro. *Critical loss: Let's tell the whole story*, ANTITRUST 17 (2002) 49.

603. For example, a product with an own-price demand elasticity of two would see sales drop by two percent for every one-percent increase in price.

604. O'Brien & Wickelgren at 10, Equation 4.

Above, *Margin* gives the hypothetical monopolist's initial price-cost margin at the competitive price. For example, if the competitive margin is 25 percent, a hypothetical monopolist would have a critical loss of (0.05/(0.25+.05)) = 0.167, or 16.7 percent. If the hypothetical monopolist raised its price by five percent above the competitive level, the price increase would be profitable so long as the hypothetical monopolist lost no more than 16.7 percent of its sales.

302. Second, an economist would then estimate the *actual* losses the hypothetical monopolist would face if it raised prices by five percent. To calculate actual losses, an economist estimates the hypothetical monopolist's own-price elasticity of demand for the products in question, as well as the cross-price elasticity of demand for other products owned by the monopolist (if any) that a consumer might switch to in response to a price increase on the first set of products. For a five percent price increase, actual loss is calculated as:

$$Actual\ Loss = 5\% * (\varepsilon_{Own} - \varepsilon_{Cross})$$

For example, a firm with an own-price elasticity of two and no cross-price elasticity would face an actual loss of (.05*2) = 0.1, or 10 percent. The cross-price elasticity is irrelevant here since Google does not own another app store aside from the Play Store (or another payment processor aside from Google Play Billing). Hence, I conservatively set ε_{Cross}= 0.

303. Finally, an economist compares the critical loss and the actual loss. So long as the actual loss is below the critical loss, the price increase would be profitable to the hypothetical monopolist. If the price increase is profitable, it implies that the products controlled by the hypothetical monopolist do not have sufficient outside substitutable alternatives to defeat an exercise of market power. The SSNIP test would therefore indicate that the products in question represent a relevant antitrust market.

304. Economists may use a SSNIP test prospectively, such as in merger cases where a SSNIP test is used to analyze whether a merged firm could profitably rise prices and could harm consumer welfare. The SSNIP test can also be used retroactively, including in cases where a not-so-hypothetical monopolist already commands the markets in question, as is the case here.[605] A retroactive SSNIP test has the potential to underestimate a firm's market power, since the prices and margins in the actual world may be contaminated by the Challenged Conduct.[606]

305. To perform a one-sided SSNIP test for the In-App Aftermarket, I start by empirically estimating Google's actual-world own-firm demand elasticity for the In-App Aftermarket. I used standard multiple regression methods. Specifically, I regressed the natural logarithm of quantity demanded of In-App Content on the natural logarithm of Google's take rate in dollars (the price Google charges for facilitating the transaction). I include App-specific fixed effects and uses taxes as an exogenous instrument. This yields an own-firm demand elasticity of

605. OECD Policy Roundtable, *Market Definition 2012,* Organisation for Economic Co-operation and Development (October 11, 2012) at 40, *available at* www.oecd.org/daf/competition/Marketdefinition2012.pdf ("In monopolisation cases or in cases of an abuse of a dominant position, the potential anticompetitive effects may already have occurred. As a result, the analysis may be retrospective and the prevailing price may already be higher as compared to the but-for price.").

606. *Id.* This is known as the "Cellophane Fallacy," named after the US Supreme Court case in *United States v EI du Pont de Nemour and Co*. Briefly, the court "fail[ed] to recognise that the prices on which the market definition was based were already tainted by the infringement. In other words, the market definition proceeded prospectively, as if in a merger case, not taking the (at least partial) retrospective effects of the alleged infringement into account."

1.88 at Google's present market share of 97 percent, implying a market demand elasticity of 1.88*0.97 =1.82. This yields an estimate of the demand elasticity that would be faced by a hypothetical monopolist with a market share of 100 percent in the In-App Aftermarket.[607] I note that this independent econometric estimate is very much in line with the market demand elasticity of 1.50 used in my economic model of In-App Aftermarket, summarized in Table 5 above. For purposes of the SSNIP test, I conservatively employ the greater of these two estimates (1.82 rather than 1.50).[608]

306. Actual loss is calculated using a SSNIP of five percent, multiplied by the market demand elasticity:

$$Actual\ Loss = 5\% * 1.88 * 97\% \ = 9.1\%$$

A hypothetical monopolist in the In-App Aftermarket could profitably raise prices above competitive levels as long as the actual loss is less than the critical loss. This will be the case as long as the competitive margin is less than 50 percent.[609] As demonstrated in my economic model of the In-App Aftermarket, summarized in Table 5 above (Row 15), Google's competitive margin ([redacted] percent) is substantially below [redacted] percent. As a consequence, the critical loss is [redacted] percent, which substantially exceeds the actual loss of [redacted] percent. Moreover, the [redacted] percent estimate of the competitive margin is conservatively high, given that the analysis in Table 5 assumes that Google enjoys brand loyalty conferring a substantial market share of 60 percent even in the competitive world. If the competitive margin were lower, the critical loss would be still higher. For example, if the competitive margin were ten percent, the critical loss would be [redacted] percent, which again substantially exceeds the actual loss of [redacted] percent. Because the actual loss is below the critical loss, a hypothetical monopolist could profitably impose a SSNIP above competitive levels in the In-App Aftermarket.

607. The market demand elasticity is given by the equation $E_g = \frac{E_M}{S_g} + \frac{E_S(1-S_g)}{S_g}$. (Landes & Posner at 944-945.) In the actual world, $E_s = 0$ because competitive rivals are constrained by Google's tie. Therefore, $E_M = E_gS_g$. *See, e.g.*, MICHAEL KATZ AND HARVEY ROSEN, MICROECONOMICS 3rd ed. 329-330 (Irwin/McGraw-Hill 1998).

608. Using the higher estimate is conservative because a greater market elasticity yields a greater actual loss. All else equal, this makes it less likely that a SSNIP would be profitable, because the SSNIP is profitable when the critical loss exceeds the actual loss.

609. Setting critical loss equal to actual loss, we obtain: 5%/(*Margin* + 5%) = 9.1%. The equation is balanced when *Margin* = 50%. The actual loss of 9.1 percent will be less than the critical loss as long as *Margin* < 50%.

APPENDIX 6: DAMAGES BY STATE

TABLE A6.1: BY STATE COMBINED ANDROID APP DISTRIBUTION AND IN-APP AFTERMARKET DAMAGES (8/16/2016 – 12/31/2020)

State (or Territory)	% of Consumer Expenditure	Damages
Alabama	[redacted]	$67,524,945
Alaska		$28,927,282
American Samoa		$92,917
Arizona		$105,883,933
Arkansas		$42,205,868
California		$534,985,767
Colorado		$90,751,900
Connecticut		$39,740,944
Delaware		$22,737,824
District of Columbia		$8,230,541
Federated States of Micronesia		$5,716
Florida		$266,864,458
Georgia		$132,621,133
Guam		$936,145
Hawaii		$27,759,379
Idaho		$28,100,254
Illinois		$171,681,928
Indiana		$104,417,051
Iowa		$52,057,981
Kansas		$48,767,475
Kentucky		$56,347,257
Louisiana		$66,826,705
Maine		$17,984,033
Marshall Islands		$30,968
Maryland		$85,378,776
Massachusetts		$79,143,988
Michigan		$156,664,683
Minnesota		$90,403,301
Mississippi		$34,747,285
Missouri		$100,034,673
Montana		$22,075,786
Nebraska		$29,802,955
Nevada		$52,636,116
New Hampshire		$20,778,372

State (or Territory)	% of Consumer Expenditure	Damages
New Jersey	[redacted]	$99,996,721
New Mexico		$31,120,940
New York		$268,913,517
North Carolina		$121,891,348
North Dakota		$16,120,218
Northern Mariana Islands		$124,657
Ohio		$175,124,607
Oklahoma		$70,787,568
Oregon		$136,961,770
Palau		$39,658
Pennsylvania		$176,498,595
Puerto Rico		$9,601,747
Rhode Island		$14,143,094
South Carolina		$64,787,397
South Dakota		$13,855,680
Tennessee		$92,563,379
Texas		$369,634,808
Utah		$45,572,143
Vermont		$7,195,792
Virgin Islands		$1,189,157
Virginia		$116,419,732
Washington		$144,911,525
West Virginia		$24,889,498
Wisconsin		$92,130,489
Wyoming		$12,848,067
Armed Forces Europe, the Middle East, and Canada		$2,968,938
Armed Forces Pacific		$2,564,549
Armed Forces Americas		$504,499
Not Listed		$25,978,079
Total		**$4,726,486,511**

Source: Google App Transaction Data; Table 10.

TABLE A6.2: BY STATE COMBINED ANDROID APP DISTRIBUTION DAMAGES (8/16/2016 – 12/31/2020)

State (or Territory)	% of Consumer Expenditure	Damages
Alabama	[redacted]	$254,693
Alaska		$79,460
American Samoa		$253
Arizona		$431,481
Arkansas		$167,021
California		$2,246,252
Colorado		$412,723
Connecticut		$162,220
Delaware		$58,177
District of Columbia		$37,801
Federated States of Micronesia		$60
Florida		$1,095,550
Georgia		$554,474
Guam		$2,672
Hawaii		$79,191
Idaho		$133,446
Illinois		$672,772
Indiana		$391,283
Iowa		$194,305
Kansas		$189,167
Kentucky		$248,667
Louisiana		$224,993
Maine		$82,349
Marshall Islands		$86
Maryland		$325,325
Massachusetts		$349,292
Michigan		$592,117
Minnesota		$367,639
Mississippi		$115,456
Missouri		$386,419
Montana		$79,858
Nebraska		$114,140
Nevada		$180,994
New Hampshire		$90,625
New Jersey		$401,360

New Mexico	[redacted]	$143,330
New York		$1,066,070
North Carolina		$505,630
North Dakota		$58,840
Northern Mariana Islands		$494
Ohio		$667,786
Oklahoma		$255,488
Oregon		$383,740
Palau		$58
Pennsylvania		$682,152
Puerto Rico		$44,885
Rhode Island		$53,593
South Carolina		$243,743
South Dakota		$55,426
Tennessee		$360,572
Texas		$1,614,763
Utah		$256,249
Vermont		$36,742
Virgin Islands		$2,653
Virginia		$487,459
Washington		$592,033
West Virginia		$105,360
Wisconsin		$346,969
Wyoming		$46,991
Armed Forces Europe, the Middle East, and Canada		$14,441
Armed Forces Pacific		$10,413
Armed Forces Americas		$2,363
Not Listed		$4,115
Total		**$18,764,677**

Source: Google App Transaction Data; Table 10.

TABLE A6.3: BY STATE IN-APP AFTERMARKET DAMAGES (8/16/2016 – 12/31/2020)

State (or Territory)	% of Consumer Expenditure	Damages
Alabama	[redacted]	$67,283,442
Alaska		$28,882,681
American Samoa		$92,779
Arizona		$105,441,508
Arkansas		$42,039,380
California		$532,619,029
Colorado		$90,287,527
Connecticut		$39,574,346
Delaware		$22,711,266
District of Columbia		$8,187,690
Federated States of Micronesia		$5,619
Florida		$265,733,374
Georgia		$132,039,120
Guam		$934,502
Hawaii		$27,710,745
Idaho		$27,945,247
Illinois		$171,017,849
Indiana		$104,048,688
Iowa		$51,875,863
Kansas		$48,582,688
Kentucky		$56,073,998
Louisiana		$66,641,432
Maine		$17,890,895
Marshall Islands		$30,918
Maryland		$85,066,887
Massachusetts		$78,760,134
Michigan		$156,101,982
Minnesota		$90,027,064
Mississippi		$34,653,992
Missouri		$99,658,825
Montana		$22,003,597
Nebraska		$29,692,934
Nevada		$52,482,684
New Hampshire		$20,679,736
New Jersey		$99,591,064
New Mexico		$30,958,127

New York	[redacted]	$267,848,970
North Carolina		$121,364,332
North Dakota		$16,066,461
Northern Mariana Islands		$124,164
Ohio		$174,483,892
Oklahoma		$70,557,247
Oregon		$136,735,672
Palau		$39,699
Pennsylvania		$175,834,734
Puerto Rico		$9,550,197
Rhode Island		$14,092,019
South Carolina		$64,556,925
South Dakota		$13,799,843
Tennessee		$92,209,620
Texas		$367,874,956
Utah		$45,241,688
Vermont		$7,150,998
Virgin Islands		$1,188,541
Virginia		$115,907,387
Washington		$144,303,022
West Virginia		$24,777,690
Wisconsin		$91,802,042
Wyoming		$12,805,035
Armed Forces Europe, the Middle East, and Canada		$2,951,882
Armed Forces Pacific		$2,553,908
Armed Forces Americas		$501,782
Not Listed		$26,071,517
Total		**$4,707,721,834**

Source: Google App Transaction Data; Table 10.

REDACTED VERSION

Exhibit A3
to
C. Cramer Declaration

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

IN RE GOOGLE PLAY STORE ANTITRUST LITIGATION THIS DOCUMENT RELATES TO: *In re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD	No. 3:21-md-02981-JD **CLASS CERTIFICATION REPLY REPORT OF HAL J. SINGER, PH.D.** Judge: Hon. James Donato

HIGHLY CONFIDENTIAL UNDER PROTECTIVE ORDER

INTRODUCTION AND ASSIGNMENT

1. I have been asked by counsel for Consumer Plaintiffs to respond to the expert report of Dr. Michelle Burtis ("Burtis Report").[1] As detailed below, the Burtis Report does not cause me to alter any of the opinions expressed in my class certification report ("Singer Report").[2]

2. In the Singer Report, I concluded that common methods and evidence demonstrate that, in the absence of Google's anticompetitive conduct, all or almost all Consumer Class members would have paid lower prices for Apps and In-App Content.[3] I demonstrated that Google has monopoly power in the Android App Distribution Market, gained and maintained through exclusionary contracting practices and artificial technological barriers.[4] I further demonstrated that Google has extended its power into the In-App Aftermarket by tying it to the Android App Distribution Market using the Aftermarket Restrictions.[5]

3. In the Singer Report, I used standard economic methods, data, and evidence common to the Class to show that the Challenged Conduct resulted in antitrust injury to all or almost all Class members, with aggregate damages of up to $4.73 billion for consumers nationwide across the two separate product markets.[6] I also presented an alternative model of impact and damages that can be applied to a single, combined antitrust product market, including one variation that does not calculate a take-rate reduction, but instead finds common impact through direct consumer subsidies.[7] Dr. Burtis is therefore mistaken when she claims that "Consumer Plaintiffs' claims of classwide antitrust impact depend on showing that there would be a reduction in service fees..."[8]

4. I have also been asked to respond to both Dr. Burtis's and Dr. Michael Williams's analysis of pass-through.[9] The pass-through analyses of Drs. Burtis and Williams are unreliable and biased, and they are virtually guaranteed to underestimate pass-through in a competitive but-for world with steering and discounting—conditions that were absent in their "natural experiments." Although their pass-through analyses suffer from numerous methodological flaws that render both of their analyses unreliable and uninformative, it bears emphasis that no "corrections" to their methods could produce reliable pass-through estimates, given (1) the limitations that Google placed on a developer adjusting its prices, (2) the lack of any incentive to rebate prices to renewing

---

1. Expert Report of Dr. Michelle M. Burtis (March 31, 2022) [hereafter, Burtis Report].
2. Expert Report of Hal J. Singer, Ph.D. (February 28, 2022) [hereafter, Singer Report]. Unless otherwise defined, capitalized terms herein are defined the same as they are in the Singer Report. The materials I relied upon in forming my opinions are noted in the footnotes throughout this report or otherwise listed in Appendix 1, or in the Singer Report. I reserve the right to supplement, expand, or amend my opinions.
3. *Id.* ¶2. I also concluded that Consumer Plaintiffs would also have benefitted from enhancements to output, quality, and consumer choice in a more competitive but-for world. *Id.*
4. *Id.* Dr. Burtis accepts for purposes of her analysis that "the alleged relevant antitrust markets are properly defined," that "Google Play had market power in the[] claimed relevant markets," and that "in the but-for world, Google Play would face more competition." Burtis Report ¶¶42-44.
5. Singer Report ¶2.
6. *Id*. ¶274.
7. *Id.* ¶¶245-256.
8. Burtis Report ¶11.
9. *Id*. ¶¶173-178; ¶¶280-295; Expert Report of Michael Williams (February 28, 2022) [hereafter, Williams Report] ¶¶76-88; ¶¶111-124. Dr. Williams is one of the economic experts for the Developer Class.

subscribers, and (3) that no data exist to directly measure pass-through in the absence of the Challenged Conduct, which remains in place to this day.[10] In addition, classwide impact does not depend on pass-through in my economic model that finds common impact through direct consumer subsidies.[11] Dr. Burtis's and Dr. Williams's pass-through analyses are irrelevant to that model, which itself supports class certification in this case.

QUALIFICATIONS

5. My qualifications are detailed in the Singer Report. Since that time, I have testified before the House Economic Disparity and Fairness in Growth Committee on April 6, 2022, regarding the linkages between concentration and consumer harms.[12]

I. DR. BURTIS'S ATTEMPTS TO REFUTE COMMON IMPACT ARE WITHOUT MERIT

6. This section responds to Part VI of the Burtis Report. Sub-section A responds to Dr. Burtis's arguments in her Part VI.A, sub-section B responds to Dr. Burtis's arguments in her Part VI.B, and so on.

A. Google Would Have Offered Lower Take Rates To All or Almost All Developers in a More Competitive But-For World

7. In Part VI.A of the Burtis Report, Dr. Burtis claims there is no common proof of a "Uniform Lower Service Fee" in the but-for world. My analysis does not *assume* a uniform take rate across all developers in a more competitive but-for world, but instead *solves* for a but-for *headline* rate that applies to almost all developers, which tracks Google's standardized pricing in the actual world. As explained in the Singer Report, Google did offer limited discounts to the headline rate of 30 percent for an exceedingly small share of developers. However, even for these developers, the take rate net of any discounts would have been lower in the but-for world because the discounts would have been negotiated from a lower headline rate, not the inflated 30 percent headline rate.[13] Therefore, in the but-for world, it is reasonable to assume that those developers that secured discounts off the inflated rate of 30 percent would have also secured the discounts off a lower, headline rate. My model determines the headline rate that would have prevailed in a more competitive but-for world, and implies antitrust impact to the small fraction of developers that obtained a discount off the headline rate.

---

10. *See* Part IV, *infra*.
11. Singer Report ¶¶245-256.
12. "(Im)Balance of Power: How Market Concentration Affects Worker Compensation and Consumer Prices," Testimony to the House Committee on Economic Disparity and Fairness in Growth, Apr. 6, 2022.
13. Singer Report ¶177, n. 376; *see also* Singer Report Part VII (demonstrating how damages can be calculated for individual Class Members using common methods, taking into account the fact that a limited number of developers received discounts relative to Google's standard 30 percent take rate).

1. Limited Variation Within the Play Store's Highly Uniform and Formulaic Take Rate Structure Does Not Refute Classwide Impact

8. In Part VI.A.1 of the Burtis Report, Dr. Burtis claims that take-rate reductions would be individualized in a more competitive but-for world. [14] The Play Store's take rate structure is highly uniform and formulaic. Dr. Burtis emphasizes limited exceptions for special programs such as LRAP and SwG (each of which offered a 15 percentage point reduction from the headline take rate), or LRAP++ (offering a 22.5 percentage point reduction from the headline take rate).[15] But only *one tenth of one percent* of the proposed developer class qualified for any of these special programs.[16] And within each of these special programs, the take rate remained highly formulaic—set equal to a fixed percentage-point discount off the headline take rate of 30 percent.

9. For the duration of the Class Period, the Play Store has charged a headline take rate of 30 percent to the vast majority of developers.[17] Starting in 2018, Google implemented limited, formulaic, non-individualized take rate reductions. In 2018, Google reduced take rates from 30 to 15 percent for developers whose customers maintained a subscription service for twelve or more continuous months; the take rate for the first twelve months on subscriptions remained at the headline rate of 30 percent. Starting in January 2022, Google reduced the take rate to 15 percent for all subscription developers, even for the initial twelve months.[18] Google applied the same common formula to all or almost all subscription developers.[19] In mid-2021, Google reduced the take rate to 15 percent for the first $1 million in annual revenue for all developers in the Play Store, again applying a common formula.[20]

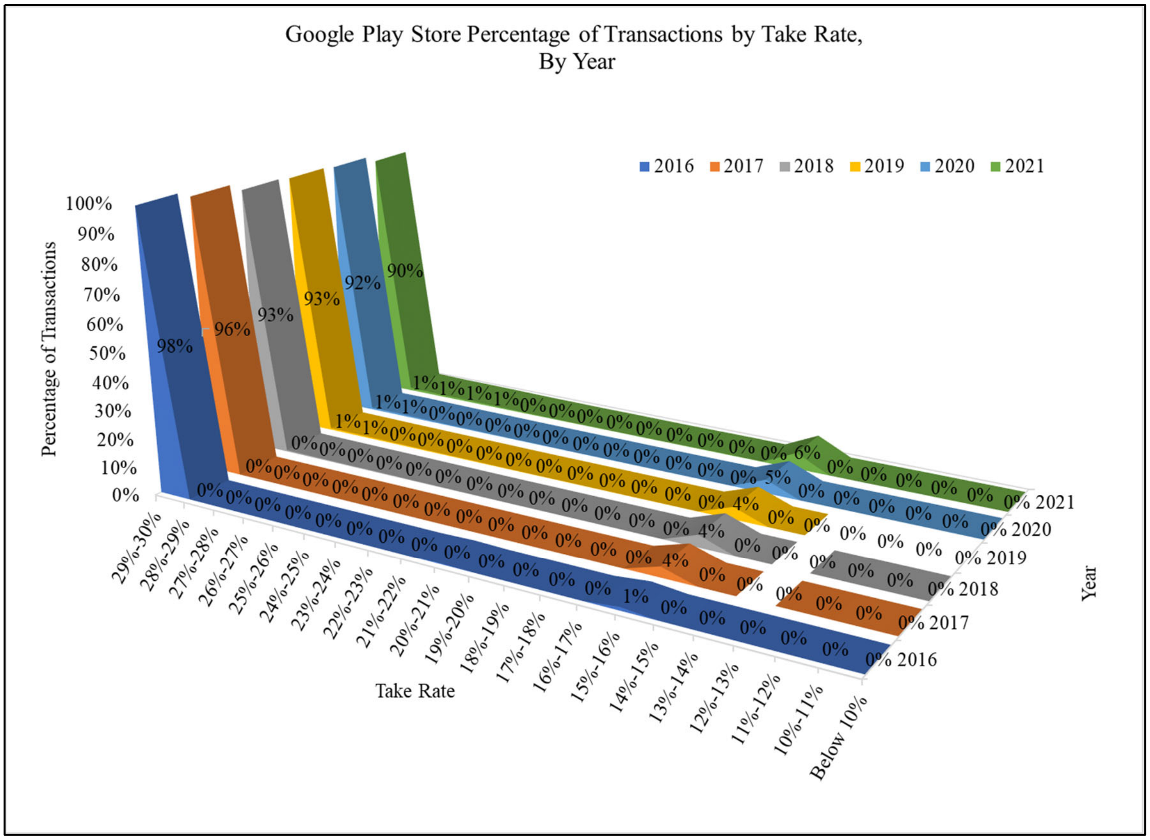
10. It is not reasonable to presume, as Dr. Burtis does, that Google would alter its pricing policy and start negotiating take rates on an individualized basis with tens of thousands of developers

---

14. Burtis Report ¶113.
15. *Id.* ¶114.
16. There are "at least 49,000 U.S. developers" in the developer class. Burtis Report ¶74. There are "[redacted] putative developer class members that participate in the programs LRAP, LRAP++, SwG, ADAP, and Hug/GVP." *Id.* ¶117. Accordingly, approximately [redacted] = [redacted] percent of proposed developer class members qualified for these special programs.
17. *See, e.g.,* GOOG-PLAY-000443763, at -772 (summarizing the Play Store's business model as charging a 30 percent take rate). *See also* Figure 1, *infra*; *see also* Singer Report Table 3, row [6] (showing [redacted] percent take rate in the Android App Distribution Market). *Id.* Table 5, row [3] (showing a [redacted] percent take rate in the In-App Aftermarket). Special programs such as LRAP were in place before 2018. As noted above, only [redacted] percent of the developer class qualified for any of these special programs.
18. Sameer Samat (Google Vice President, Product Management), *Evolving our business model to address developer needs* (Oct. 21, 2021), *available at* https://android-developers.googleblog.com/2021/10/evolving-business-model.html ("starting on January 1, 2022, we're decreasing the service fee for all subscriptions on Google Play from 30% to 15%, starting from day one.")
19. A possible exception would be a subscription developer that was among the [redacted] percent of developers that qualified for special programs such as LRAP.
20. Sameer Samat (Google Vice President, Product Management), *Boosting developer success on Google Play* (Mar. 16, 2021), *available at* https://android-developers.googleblog.com/2021/03/boosting-dev-success html ("Starting on July 1, 2021 we are reducing the service fee Google Play receives when a developer sells digital goods or services to 15% for the first $1M (USD) of revenue every developer earns each year. With this change, 99% of developers globally that sell digital goods and services with Play will see a 50% reduction in fees.").

in a more competitive but-for world.[21] Figure 1 below illustrates the uniformity of the Play Store's take rate structure over the Class Period, showing the vast majority of take rates between [redacted] and 30 percent. Figure 2 focuses on take rates above [redacted] percent, and shows that the vast majority of these are above [redacted] percent.

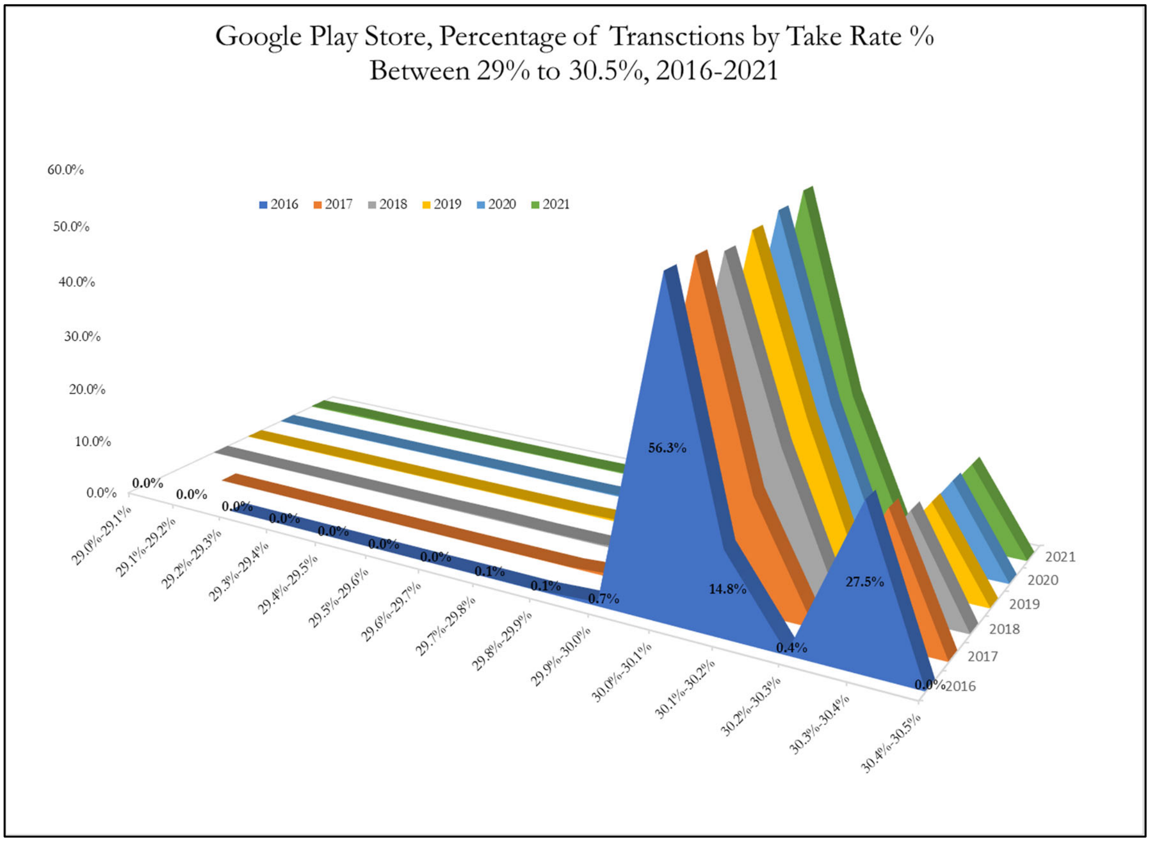
FIGURE 1: THE PLAY STORE'S UNIFORM AND FORMULAIC TAKE RATE STRUCTURE



Note: Figures may not sum to 100 percent due to rounding. Take rates of 15 percent before 2018 reflect special programs such as LRAP, as well as [redacted], which has largely circumvented the Aftermarket Restrictions. Singer Report ¶154.

21. To secure promises by certain of the largest developers not to provide other app stores (like Samsung) with exclusive content, Google negotiated certain benefits to those developers as part of "Project Hug." This program was limited to a small number of developers. Even during the Project Hug negotiations, Google did not offer individualized take rates. Singer Report ¶¶110-114; ¶¶121-127.

FIGURE 2: TAKE RATES ABOVE 29 PERCENT IN THE PLAY STORE



Notes: Figures may not sum to 100 percent due to rounding. Also due to rounding, Google's calculated take rate can exceed 30 percent for some lower-priced transactions. Singer Report ¶193, n. 392.

11. Dr. Burtis claims that, because entry by competing app stores such as the ONE Store and the Epic Games Store did not cause Google to lower its take rate in the actual world, this demonstrates that all or almost all Class members would not have benefitted from lower take rates in a more competitive but-for world.[22] Yet Google *did* respond to the (limited) competition offered by the ONE Store by expanding subsidies available to all consumers via its Play Points program in South Korea.[23] As Dr. Burtis concedes, "Google Play Points are available, without cost, to any consumer that registers to receive Play Points."[24] And the Epic Games Store was not even available on Android devices (only on PC and Mac), as Dr. Burtis concedes.[25] That Google has not responded

---

22. Burtis Report ¶¶128-129.
23. Singer Report ¶248.
24. Burtis Report ¶353.
25. *Id.* ¶129; *see also* Epic Games Store, *FAQ*, *available at* https://www.epicgames.com/site/en-US/epic-games-store-faq ("Which Platforms does the Epic Games Store Support? The Epic Games Store currently offers PC and Mac support.") (last updated Aug. 18, 2021).

further to competition from competing app stores is indicative of the lack of full-fledged competition in the actual world.[26]

12. Dr. Burtis claims that, because competing app stores such as those operated by Samsung and Amazon have offered "special incentives" to certain developers, this demonstrates that individualized inquiry would be necessary to determine whether all or almost all developers would have benefitted from lower take rates in the but-for world.[27] But she concedes that Samsung offered a formulaic "tiered structure," with take rates of either 30 percent or 20 percent for "premier partners."[28] That competing app stores agreed to lower take rates for exceptionally popular developers such as Epic provides no basis to presume, as Dr. Burtis does, that competing app stores would negotiate take rates on an individualized basis with tens of thousands of developers in a more competitive but-for world. More importantly, Samsung's and Amazon's "targeted incentives" did not amount to meaningful competition in light of the Challenged Conduct, as explained in the Singer Report.[29] That Google did not drop its take rate in response to this weak entry does not inform how Google would have responded to vibrant entry in the but-for world with steering and discounting.

2. Developers at Low Price Points Would Still Benefit from Increased Competition in a More Competitive But-For World and Would Not Pay Higher Payment Processing Costs as a Percentage of the Transaction

13. This section responds to Part VI.A.2 of the Burtis Report, in which Dr. Burtis claims that developers with low price points ($0.99 per download) would not have benefitted from increased competition because payment processors such as PayPal currently impose a per-transaction fee of up to $0.49, resulting in payment processing fees in excess of 50 percent for developers charging $0.99.[30] Dr. Burtis also claims these per-transaction fees would result in take rates in excess of 27 percent for developers charging $1.99.[31] It bears noting that the 27 percent implied rate is *below* what Google charged for most of the Class Period, indicating that these developers suffered antitrust injury, even if one accepts Dr. Burtis's assumption that a per-transaction fee would persist in the but-for world.

14. Dr. Burtis's claim that the competitive rate for payment processing would range from 27 to 50 percent wholly ignores the realities of current pricing of transactions for paid Apps and In-App Content. Google obviously does not incur a per-transaction fee of $0.49 each time it processes

---

26. *See, e.g.,* GOOG-PLAY-000005203.R at GOOG-PLAY-000005221.R (Google characterizes ONE Store as "strictly a [Korean] channel," with "low [Monthly Average Users]" and a "very poor catalog" of apps); GOOG-PLAY-000292207.R at GOOG-PLAY-000292223.R (Google contrasts Play as the "everything to everyone" store, while ONE Store "can only target narrow slices of our users,").

27. Burtis Report ¶¶119-121.

28. *Id*. ¶120. With respect to Amazon, the document cited in the Burtis Report (¶119, n. 32) shows that Google recognizes that Amazon does not provide the competitive discipline that would characterize a more competitive but-for world. *See* GOOG-PLAY-000568027.R at 028.R ("Amazon [has] yet to establish critical mass.").

29. *See* Singer Report Part III.D.2.c (Google Discouraged Samsung from Effectively Competing with the Play Store in the Distribution of Apps in the Android App Distribution Market and Entered into Deals with Developers to Mitigate the Risk of Competition from Samsung); *see also* Part III.D.2.b (Google Has Deployed Multiple Measures to Ensure That Amazon Would Not Become an Effective Play Store Competitor).

30. Burtis Report ¶¶131-132.

31. *Id.*

a $0.99 transaction for a developer (if it did, it would lose money on these developers). Record evidence [redacted] serve $0.99 developers while charging much lower take rates than the 52 percent payment processing costs that Dr. Burtis claims developers would incur in a more competitive but-for world.[33] Google and other app stores achieve economies of scale in payment processing by aggregating transactions from a large number of developers—economies that have been denied to payment processors in part due to the Challenged Conduct. In a more competitive but-for world, absent the Aftermarket Restrictions, competing payment processors would gain a foothold in the In-App Aftermarket and enjoy these economies of scale and, if they wanted to win the business of apps pricing at $0.99, economics predicts that they would eschew minimum fees—indeed, this competitive process has already begun to unfold in the Apple App Store, as explained below.

15. In Table 4 of her report, Dr. Burtis lists *some* of the fee structures that payment processors currently advertise in distinct markets and claims that developers transacting at low prices such as $0.99 would be made worse off under these rate structures, paying take rates in excess of even Google's 30 percent.[34] Dr. Burtis speculates that "a developer with a $0.99 price that used PayPal would pay $0.49 plus 2.59% of the price. The total cost is $0.52 or 52% of the value of the transaction – substantially higher than Google Play's 30% (or 15%) rate."[35] But Dr. Burtis fails to disclose that the same payment processors she cites offer "micropayment" fee structures specifically tailored to small-dollar transactions—another fact she ignores in her analysis. For example, PayPal has long advertised and offered micropayment rates of five percent plus a fixed fee of $0.05.[36] For a $1 transaction, the payment processing fee would come to just ten percent—far below Google's 30 percent take rate, let alone the rates in excess of 50 percent that Dr. Burtis puts forward.[37] Thus, Dr. Burtis's speculation that "developers that rely primarily on low priced app transactions" would

---

32. *See, e.g.*, GOOG-PLAY-001556407 at -409 [redacted] *See also* GOOG-PLAY-007879368.C at GOOG-PLAY-007879369.C ("all [redacted] transactions [redacted] processed at [redacted] [.]")

33. Burtis Report ¶132.

34. *Id.*

35. *Id.*

36. *See* Paypal Help Center, *How can I update my payment preferences for micropayments?*, *available at* https://www.paypal.com/gf/smarthelp/article/how-can-i-update-my-payment-preferences-for-micropayments-faq1691 ("Our micropayment pricing is designed for PayPal Business or PayPal Premier customers who process transactions that are typically under $12 USD. As long as you have a Business or Premier PayPal account, you can enjoy these rates. To update or apply for the micropayments pricing, give us a call by clicking Help at the top of the page and we will assist you further. Our micropayment rate is 5% (6% for crossborder transactions) + $0.05 USD per transaction."); GOOG-PLAY-008162331 (2021 Google spreadsheet showing Paypal micropayment rate of $0.05 plus 5%). As of August 2, 2021, it appears that Paypal has increased its Micropayments fee to $0.09 plus 4.99%. *See* PayPal Fee Changes, (Aug. 2, 2021), *available at* https://www.paypalobjects.com/marketing/ua/pdf/US/en/feepages-080221.pdf. Even at this rate, an effective rate of 14 percent for a $0.99 transaction (equal to $0.09 + $0.99 x 0.0499) is well below Dr. Burtis's estimates.

37. Burtis Report ¶132.

be forced to rely on "more expensive"[38] payment processors in a more competitive but-for world is wrong, and is contradicted by evidence in the actual world.

16. Dr. Burtis's error becomes even more apparent when evaluating the aftermath of the recent *Epic v. Apple* ruling that Apple must permit app developers to steer consumers to other payment processing systems. Following this decision, the third-party payment processing firm Paddle announced its pricing structure for Apple's App Store.[39] In addition to providing a suite of merchant services that Apple does not, Paddle offers a ten percent take rate for any transactions under $10, and a five percent take rate plus $0.50 for transactions over this amount.[40] As shown below, if Paddle were to provide the same services to developers on the Play Store at such rates, developers selling low-priced Apps (below $10) would be unequivocally better off, as they would pay only a ten percent transaction fee with no fixed component:

TABLE 1: TAKE RATES FOR APPLE APP STORE, THE PLAY STORE, AND PADDLE

	App Store	**Play Store**	**Paddle**
Transactions below $10	15-30%	15-30%	10%
Transactions from $10+	15-30%	15-30%	5% + $0.50

Source: https://www.paddle.com/platform/in-app-purchase

Indeed, even for App prices above $10, developers would be better off under Paddle (5 percent plus $0.50) than paying the lowest take rate from Google (15 percent). For example, the lowest fee for an App priced at $10.00 on the Play Store would be $1.50, whereas the fee on the same App prices at $10.00 on Paddle would be $1.00 (equal to $0.50 plus $0.50). The disparities increase with increase in the price of the App—the Play Store is always more expensive than Paddle.

17. Dr. Burtis claims that "a developer that opted out of Google Play Billing would additionally have to incur costs for the other services that Google provides, such as customer service, subscription management, and management of billing disputes."[41] In fact, Paddle already includes such features, plus many others, in its fee structure.[42]

18. The uniformity of Paddle's rate structure also undermines Dr. Burtis's claim that individualized inquiry would be necessary to determine whether developers would be better off in a more competitive but-for world: Paddle offers the same rate structure to all developers. Individualized inquiry is also not necessary to arrive at the sound economic conclusion that developers would rationally select a lower-cost payment processor, other things equal.

19. Further, Dr. Burtis ignores that not every developer has to switch to a competing payment processor to benefit in a more competitive but-for world. Indeed, my analysis conservatively assumes that only 40 percent would switch, leaving 60 percent of the In-App

38. *Id.* ¶131.

39. Chance Miller, Paddle unveils 'first alternative' to Apple's App Store In-App Purchase system following Epic ruling, 9TO5MAC, (Oct. 7, 2021), *available at* https://9to5mac.com/2021/10/07/app-store-iap-paddle-system-announcement/.

40. In-App Purchase, Paddle, *available at* https://www.paddle.com/platform/in-app-purchase.

41. Burtis Report ¶133.

42. In-App Purchase, Paddle, *available at* https://www.paddle.com/platform/in-app-purchase.

Aftermarket remaining with Google. Those that remain with Google would still benefit from lower take rates, as Google would be forced to lower its take rate to remain competitive according to standard economic principles.[43]

B. All or Almost All Class Members Would Have Paid Lower Prices in a More Competitive But-For World

20. This section responds to Part VI.B of the Burtis Report, in which Dr. Burtis claims that "Differences in cost conditions for different apps, differences in the demand elasticities for different apps, and differences in app developers' pricing strategies indicate that many developers would not charge lower retail prices for apps, subscriptions, and IAPs in a but-for world with uniformly lower service fees."[44]

1. Individualized Analysis of Developer Marginal Costs Is Not Necessary To Demonstrate Classwide Impact

21. In Part VI.B.1 of the Burtis Report, Dr. Burtis claims that some Apps have zero marginal costs, while others have positive and substantial marginal costs, and suggests that Apps with zero marginal costs are "less likely" to charge lower prices to consumers when their take rates are lower.[45] Dr. Burtis provides *no evidence* that any App actually has zero marginal cost. She would be hard pressed to come up with a relevant example, given that developers incur variable payment processing costs whenever they transact with their customers. Dr. Burtis also ignores that the take rate itself represents a marginal cost that applies to the initial App purchase and any purchase of In-App Content.[46]

22. Dr. Burtis herself identifies various additional marginal costs incurred by Apps, including customer support, processing of customer information, and other costs that naturally increase with the number of users.[47] The economic literature recognizes that "ongoing marginal costs for app developers arise from various maintenance tasks after app development."[48] Per Ghosh and Han (2014), sources of ongoing marginal costs for developers include (but are not limited to): "1)

---

43. Singer Report ¶¶213-220.

44. Burtis Report ¶139.

45. *Id.* ¶142.

46. The take rate is analogous to the royalty stack that exists in the mobile phone market; this is the aggregation of individual patent royalties paid to individual standard essential patent holders, whose intellectual property the end product (the mobile phone) includes. "Running royalties," which are royalties that increase with the number of units sold, as do developer payments to Google, are considered a marginal cost. *See, e.g.,* Alexander Galetovic, Stephen Haber, Lew Zaretzki, *An estimate of the average cumulative royalty yield in the world mobile phone industry: Theory, measurement and results*, 42(3) TELECOMMUNICATIONS POLICY 263, 264 (2018) ("A standard-compliant phone uses hundreds, if not thousands of standard essential patents (SEPs) owned by a large number of SEP holders. Each SEP holder sets her running royalty rate independently, and the result is that excessive running royalties are piled on top of excessive running royalties—a theoretical construct that is called 'royalty stacking.' This royalty stack drives up the marginal cost of manufacturing phones, thereby increasing prices to consumers, and discouraging innovation by manufacturers."). *Id.* at 266 ("As the well-known theory of vertical control shows, running royalties will ultimately show up in the marginal cost of manufacturing a phone no matter where they are charged in the value chain.").

47. Burtis Report ¶¶144-146.

48. Anindya Ghose & Sang Pil Han, *Estimating Demand for Mobile Applications in the New Economy*, 60(6) MANAGEMENT SCIENCE 1470, 1474 (2014).

fixing crashes or errors reported by users of the app, 2) adding features requested by users after release, 3) user support … and 4) scaling costs."[49]

23. Dr. Burtis claims that "if an app has zero (or close to zero) marginal cost, a reduction in the service fee rate will be less likely to lead to a change in the retail price of the app."[50] Ignoring the evidence of positive marginal costs above, if what Dr. Burtis was proposing were true, then imposing a percentage sales tax on an App should have little or no effect on its pricing. But, as explained in the Singer Report, when a digital product is subject to a sales tax, this burden is typically passed through in full to the customer.[51] More generally, Google's own data shows a positive and highly statistically significant relationship between taxes and the price of Apps and In-App Content.[52] In addition, according to Dr. Burtis's logic, developers would not charge a higher price in the Play Store than on their websites. That some do, as shown in Table 9 of the Singer Report, implies that (despite the Challenged Conduct), lower take rates have led to lower prices for these developers.

24. More fundamentally, Dr. Burtis ignores that the competitive but-for world is a long-run equilibrium, in which Google's take rate is substantially and permanently lower for all or almost all developers. Standard economics shows that a developer's decision to enter and remain in the market depends on its ability both to cover its explicit costs and to earn a competitive rate of return. A developer obligated to pay 30 percent of its revenue to Google in perpetuity needs to charge a higher price to consumers than a developer facing a substantially and permanently lower take rate; the greater the take rate, the higher the developer's price will need to be for the developer to earn a competitive rate of return.[53]

25. In summary, demonstrating classwide impact does not require individualized analysis of developer costs; what matters is that developers with substantially and permanently lower costs in the but-for world would charge lower prices to all or almost all Class members.[54] As explained in Part II.A.1 below, I have employed standard economic models that fit the data and facts of this case to quantify this pass-through.

2. Individualized Analysis of Developer Pricing Strategies Is Not Necessary to Demonstrate Classwide Impact

26. In Part VI.B.2 of the Burtis Report, Dr. Burtis claims that individualized inquiry would be necessary because many developers use a "strategy of setting retail prices that end in '99.'"[55] According to Dr. Burtis, developers would be reluctant to reduce prices to a number not

49. *Id.* at 1477.
50. Burtis Report ¶142.
51. Singer Report ¶244.
52. In the Singer Report, I performed instrumental variable ("IV") regressions using Google's app revenue metrics data. Singer Report ¶¶237-238. In the first stage of the IV regressions, taxes are used as an exogenous instrument that shifts price independently of other demand drivers. According to those first-stage regressions, there is a positive and highly statistically significant relationship between taxes and App prices, even after controlling for more than 200,000 App-level fixed effects.
53. Singer Report ¶227.
54. *Id*. Part V.D.
55. Burtis Report ¶148.

ending in \$0.99. Dr. Burtis presents statistics suggesting that 97 percent of transactions in the Play Store are at prices ending in "99,"[56] but (as before) fails to disclose that more than 20 percent of the top paid Apps in the Play Store have initial download prices that *do not* end in "99,"[57] indicating that developers are willing to prominently display such prices to consumers. More broadly, Google's transaction data records at least 130,869,000 U.S. transactions at prices that do not end in "99." This is inconsistent with Dr. Burtis's claim that developers would somehow feel compelled to charge prices ending only in "99" in the but-for world.

27. According to Dr. Burtis's logic, developers would not charge a price ending with any digits other than 99 for downloading an App on their websites. That they do, as shown in Table 9 of the Singer Report, implies that developers are willing to charge prices that violate the "99 rule" when passing on the marginal cost savings from lower take rates to their customers.

28. To illustrate, suppose that a developer has consistently charged \$1.99 for In-App Content in the actual world due, in part, to focal-point pricing. Google would have kept \$0.597 per transaction (equal to 30 percent of \$1.99), remitting the remainder to the developer. Steering is prohibited for this developer in the actual world. The gross margin on each in-app transaction (before considering other marginal costs) is \$1.39 (equal to \$1.99 less \$0.597). In a but-for world, a new payment processor (such as Paddle) emerges due to the elimination of the Aftermarket Restrictions and charges a ten percent take rate, which for this developer, would be \$0.199 at the original price of \$1.99. With steering permitted, the developer can realize cost savings of \$0.398 per transaction (equal to approximately \$0.597 less \$0.199), but only if the developer can induce its customers to use the new payment processor rather than Google Play Billing. At this point, it is now profit-maximizing for the developer to deviate from focal-point pricing and share a portion of the savings with its customer via a lower price for in-app purchases. For example, the developer could drop its price for In-App Content made via the new processor to (say) \$1.79, effectively splitting the savings with the customer for making the right choice. The developer's fee to the third-party payment processor falls to \$0.179 per transaction (equal to 10 percent of \$1.79). The developer's new gross margin (before considering other marginal costs) on transactions processed via the third-party processor is \$1.61 (equal to \$1.79 less \$0.179), which exceeds the prior gross margin of \$1.39. It no longer pays to abide by focal-point pricing. A key reason we do not see such deviations from focal-point pricing in the actual world is that developers are not afforded the opportunity to steer due to the Aftermarket Restrictions. Moreover, in the few episodes where we do observe steering in the actual world, developers have been observed to deviate from 99-cent pricing increments, as shown in Table 9 of the Singer Report.

29. Dr. Burtis also ignores that Google *required* developers, until very recently, to charge at least 99 cents. Thus, for a large number of developers, the observed prices in the actual world could be the result of a restraint that Google imposed on their pricing. Record evidence indicates

---

56. *Id*. Figure 7.

57. Of the 200 "Top paid apps" in the Play Store, 43 of them (or 21.5 percent) have initial download prices that do not end in "99." *See* backup materials to this report. *See also* Top Paid Apps, Google Play, *available at* https://play.google.com/store/apps/collection/cluster?clp=0g4jCiEKG3RvcHNlbGxpbmdfcGFpZF9BUFBMSUNBVElPThAHGAM%3D:S:ANO1ljLdnoU&gsr=CibSDiMKIQobdG9wc2VsbGluZ19wYWlkX0FQUExJQ0FUSU9OEAcYAw%3D%3D:S:ANO1ljIKVpg.

that developers requested (and ultimately received) increased pricing flexibility, eventually persuading Google to abandon its $0.99 pricing floor, so that they could deviate from $.99.[58] That Google imposed a 99-cent restriction on developers implies that Google believed developers would deviate from such pricing; if focal-point pricing was as powerful as Dr. Burtis claims, Google's restraint would have been unnecessary. I understand that Plaintiffs challenged Google's 99-cent restriction in the Complaint,[59] as undermining price competition among developers, which implies that such a restriction would be absent in the but-for world.

30. To the extent developers would prefer to maintain "supermarket-style" pricing in the but-for world, they could do so simply by ending their prices in "9," instead of "99" (e.g., $2.49), or in or in "5" (e.g. $4.95) as many do today.[60] (Thus, the solution above to my hypothetical example of steering was $1.79, twenty cents below $1.99, but still ending in a nine.) Sellers' strategy of ending prices in "9" or "5", such as $2.99 or $2.95" is commonly known as "odd pricing," "psychological pricing," or "charm pricing." The strategic reasoning that underlies this practice rests on the belief that consumers will focus on the numbers to the left of the decimal, thus demonstrating higher demand for a good priced at $2.99 than $3.00, despite the negligible price difference of one cent. The economic literature has classified odd prices to include those within 5 cents of the nearest highest dollar (.95 to .99) or one cent below the next highest ten cents (.19, .29, etc.).[61] Market practitioners sometimes apply the same heuristic, underscoring the fact that odd pricing does not limit itself only to prices ending in "99."[62] Thus, Dr. Burtis errs in her attempt to cabin this strategy to such prices. Nothing would prevent developers from setting prices at $1.79 versus $1.99, for

---

58. *See* GOOG-PLAY-000355570.R at GOOG-PLAY-000355597.R ("From the developers perspective, they want more flexibility around their pricing. Many Developers that tested sub dollar pricing find this strategy effective, and they have asked us to expand the sub dollar capability to more markets."). The Play Store ultimately removed its $0.99 minimum pricing rule in our around early 2022. Archived web pages show that the Play Store had a U.S. minimum price of $0.99 as of late 2021. Play Console Help, Supported Locations for Distribution to Google Play Users, accessed Dec. 27, 2021, *available at* https://web.archive.org/web/20211227224037/https:/support.google.com/googleplay/android-developer/answer/10532353?visit_id=637762416354084080-1400722469&rd=1. As of mid-February 2022, the minimum price is listed at $0.05. Play Console Help, Supported Locations for Distribution to Google Play Users, accessed Feb. 18, 2022, *available at* https://web.archive.org/web/20220218131358/https:/support.google.com/googleplay/android-developer/answer/10532353?visit_id=637807868385671271-2942202130&rd=1.

59. Complaint at ¶¶18, 173, *In re Google Play Consumer Antitrust Litigation*, No. 3:20-CV-05761-JD (Dec. 20, 2021).

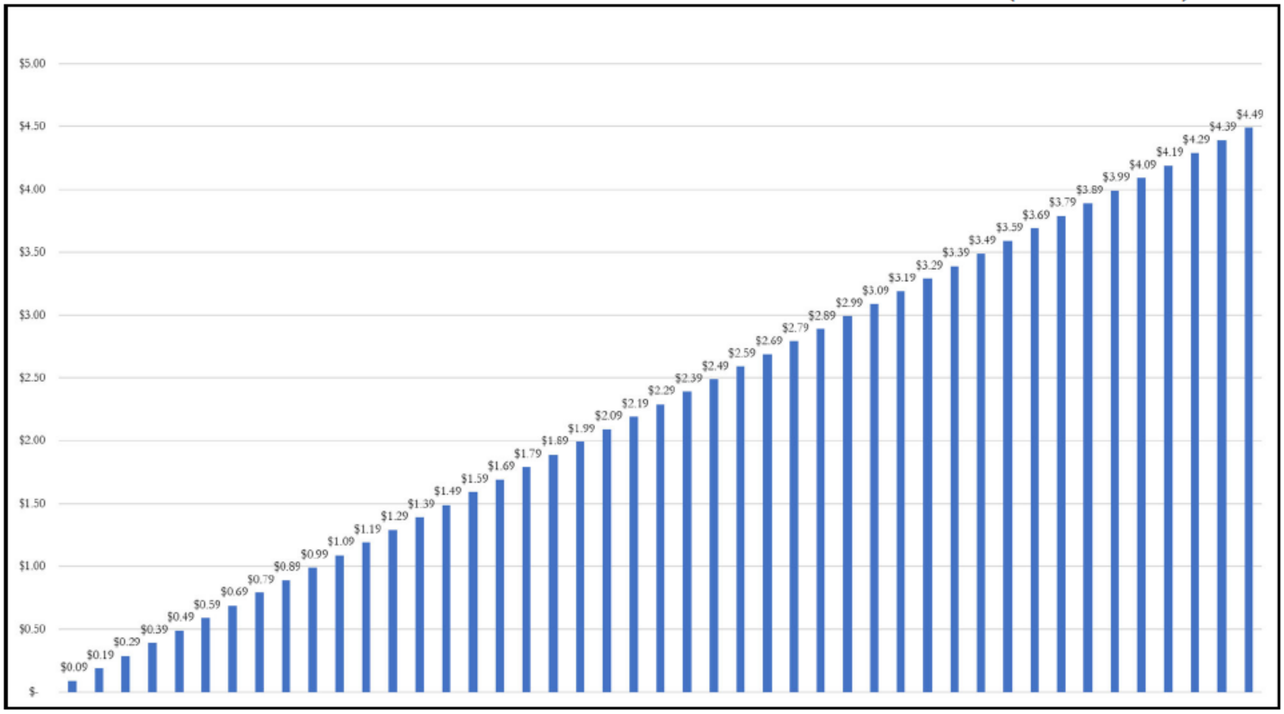
60. *See, e.g.*, Top Paid Apps, Google Play , *available at* https://play.google.com/store/apps/collection/cluster?clp=0g4jCiEKG3RvcHNlbGxpbmdfcGFpZF9BUFBMSUNBVElPThAHGAM%3D:S:ANO1ljLdnoU&gsr=CibSDiMKIQobdG9wc2VsbGluZ19wYWlkX0FQUExJQ0FUSU9OEAcYAw%3D%3D:S:ANO1ljIKVpg. (showing various "Top paid apps" with prices that do not end in "99." For example, as of April 21, 2022, the third most-popular paid App was "Torque Pro (OBD 2 & Car)," priced at $4.95 per download on the Play Store. Another paid App in the Top 20 was "Tasker," priced at $3.49 per download on the Play Store).

61. Judith Holdershaw, Philip Gendall and Ron Garland, *The Widespread Use of Odd Pricing in the Retail Sector*, MARKETING BULLETIN, 8, 1997, 53-58, Research Note 1, *available at* http://marketing-bulletin massey.ac nz/V8/MB_V8_N1_Holdershaw.pdf.

62. *See* PriceIntelligently, Odd-Even Pricing, *available at* https://www.priceintelligently.com/odd-even-pricing# ("Odd pricing refers to a price ending in 1,3,5,7,9 just under a round number, such as $0.19, $2.47, or $64.93"). *See also*, Shopify, Odd-Even Pricing, *available at* https://www.shopify.com/encyclopedia/odd-even-pricing ("Odd-even pricing is a pricing strategy involving the last digit of a product or service price. Prices ending in an odd number, such as $1.99 or $78.25, use an odd pricing strategy, whereas prices ending in an even number, such as $200.00 or 18.50, use an even strategy").

example, as a result of a decrease in the take rate. As illustrated in Figure 3 below, "supermarket pricing" is already observed in the Play Store, with prices at regular, ten-cent intervals.

FIGURE 3: PLAY STORE TRANSACTION PRICES AT TEN-CENT INTERVALS ($0.09 - $4.49)



Source: Google Transaction Data (GOOG-PLAY-007203251). Note: Prices are also observed at one-cent intervals (not shown here).

This pattern of ten-cent (or smaller) pricing increments remains consistent until price points above $100 are reached.[63]

31. In Part VI.B.2 of the Burtis Report, Dr. Burtis also claims that individualized inquiry would be necessary because "[d]evelopers use other price-setting strategies, besides short-term profit maximization."[64] Dr. Burtis ignores that the basic economic logic of pass-through applies both to short-run and long-run profit maximization. In the short run, positive marginal costs are sufficient to generate pass-through given a change in the take rate (and Dr. Burtis has not shown that any developer faces zero marginal costs). Moreover, to remain in business over the long run, a firm must recover not just its marginal costs, but its recurring fixed costs, in addition to covering its cost of capital.[65] All else equal, developers facing higher take rates must charge higher prices to remain in business. Consumers will therefore pay lower prices in a more competitive but-for world, regardless of whether one assumes that developers are maximizing profit over the short run or the long run.

63. Google Transaction Data (GOOG-PLAY-007203251).
64. Burtis Report ¶¶153-154.
65. Singer Report ¶¶223-226; *see also* Part I.B.1 above.

3. Individualized Analysis of Competitive Conditions Is Not Necessary to Demonstrate Classwide Impact

32. In Part VI.B.3 of the Burtis Report, Dr. Burtis claims that individualized analysis is necessary because developers facing different demand elasticities (that is, different competitive conditions) might not cut their prices by the same amount in the but-for world.[66] Dr. Burtis ignores that, although the extent of pass-through to consumers of a reduction in a developer's costs may vary with competitive conditions, the fact of pass-through does not. For example, as explained in the Singer Report, even a profit-maximizing monopolist will pass on a portion of cost savings to consumers.[67] Moreover, as explained in the Singer Report and in Part II.A.1 below, my pass-through analysis allows me to quantify the extent of pass-through using standard economic methods that account for differences in the competitive conditions facing different developers: I use a standard logit demand systems frequently employed by economists seeking to model pricing decisions among differentiated firms.[68] This provides a standard, tractable economic framework to estimate equilibrium pass-through rates for thousands of developers making thousands of interdependent pricing decisions for each of their Apps and In-App Content.

C. Dr. Burtis Fails to Identify Additional Economic Reasons Class Members Were Uninjured

33. In Part VI.C of the Burtis Report, Dr. Burtis claims to offer additional "economic reasons some members of the putative consumer Class would not be better off in the but-for world."[69]

1. Dr. Burtis Fails to Identify Customers Who Would Not Have Paid Lower Retail Prices in the But-For World

34. In Part VI.C.1 of the Burtis Report, Dr. Burtis claims that some consumers were not impacted because "[c]ertain retail prices are the same in Google Play as prices on the relevant developer's website."[70] This ignores that the Challenged Conduct removes the full incentives that would be present for developers to steer to their website for both App downloads and In-App Content. With respect to the In-App Aftermarket, Google's anti-steering restrictions explicitly prevent developers from directing customers *inside* an App downloaded from the Play Store to lower-cost options outside of the Play Store. Because they cannot steer customers from within the Play Store, developers are deprived of what is likely the most efficient channel for steering and have

---

66. Burtis Report ¶¶155-160.

67. Singer Report ¶223. The Singer Report also demonstrates that pass-through can be calculated for individual App categories using data and methods common to the Class. *Id.* Table 8.

68. *See* Part II.A.1 below; *see also* Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects,* 70 ANTITRUST LAW JOURNAL 257 (2002); Nathan Miller, Conor Ryan, Marc Remer, & Gloria Sheu, *Approximating the Price Effects of Mergers: Numerical Evidence and an Empirical Application* DOJ Economic Analysis Group Discussion Paper 12-8 (2012), at 2 (explaining logit is one of the demand systems "that commonly are employed in antitrust analysis."). Dr. Burtis concedes that logit demand is "frequently used in economics[.]" Burtis Report ¶306.

69. Burtis Report ¶¶161-167.

70. *Id.* ¶169.

less of an incentive to attempt to drive traffic to their websites.[71] Moreover, for initial App downloads in the Android App Distribution Market, Google has placed significant restrictions on users attempting to download competing Apps, including non-Google app stores.[72] These restrictions decrease the likelihood that steering would be successful in the Android App Distribution Market.[73] In light of these impediments, that some developers have not engaged in steering in the actual world does not imply that they would not do so in a more competitive but-for world.

2. Dr. Burtis Fails to Show That Consumers Transacting at "Lower Price Points" Were Not Impacted

35. In Part VI.C.2 of the Burtis Report, Dr. Burtis claims that some Class members were not impacted due to developers' tendency to select prices ending in "99."[74] I have already responded to this claim in Part I.A.2 above.

3. Dr. Burtis Fails to Show That Consumers Using Higher-Cost Payment Methods such as Direct Carrier Billing Were Not Impacted

36. Dr. Burtis claims that consumers who rely on "relatively expensive forms of payment to transact in Google Pay, such as direct carrier billing ('DCB') and gift cards" would not have been better off in a more competitive but-for world because "the cost to Google of processing DCB and gift cards is high relative to other forms of payment," and "[d]evelopers may not find lower cost options for those forms of payment in the but-for world."[75]

37. Dr. Burtis ignores that the same competitive forces that would lower prices for *lower*-cost forms of payment in the but-for world would also apply to *higher*-cost forms of payment. Dr. Burtis's logic presumes that consumers would not switch to a different payment form if they could take advantage of the lower prices that would prevail under the competitive counterfactual. Consumers who use DCB currently pay the same App price as those who do not. But in the "but-for" world, those who use other forms of payment would pay less, as developers pass through the cost savings from the reduced take rate. Payment processing via DCB would either have to fall to

---

71. Singer Report ¶¶25-26. In the actual world, some developers have attempted to engage in limited steering by setting a lower price on their websites, but this strategy depends on making consumers aware of the pricing differential outside of the Play Store. *Id*. ¶¶242-244; Table 9.

72. *Id*.¶¶129-130. For example, to download the Amazon app store in 2016, a user had to complete a series of 19 steps, including selecting "Unknown Sources" within the user's security settings, and navigating three separate security warnings. A Google presentation recognized the significance of this sideloading deterrent by documenting the 19 steps required to successfully install Amazon Underground. *See* GOOG-PLAY-000297309.R at GOOG-PLAY-00297311.R-00297314.R. *See also* Expert Report Of Douglas Craig Schmidt, PhD (February 28, 2022), Part IV.A (Google Android Forces Users To Go Through the Unknown Sources Flow To Install Apps, Including Non-Google App Stores, Outside Of The Google Play Store).

73. *See* [redacted] *See also* GOOG-PLAY-002414772 (In 2018, Google found only 15% of devices in the United States had the "unknown sources" setting enabled, which I understand allows downloads from outside of Play).

74. Burtis Report ¶¶148-152.

75. *Id*. ¶164; ¶¶183-185. According to Google's transactional data, only about [redacted] percent of transactions in the Play Store use DCB. Google Transaction Data (GOOG-PLAY-007203251).

meet the new competitive benchmark or risk customers abandoning DCB in favor of lower prices elsewhere.

38. Dr. Burtis also ignores that Google did not exclude DCB payments when it lowered the take rate to 15 percent for subscription developers or when it did the same for the first $1 million in revenue for all developers. Instead, Google implemented a uniform, formulaic take-rate reduction, regardless of payment method.[76] This real-world evidence contradicts Dr. Burtis's speculation regarding a more competitive but-for world.

4. Dr. Burtis's Claim That Free Apps Would Not Be Free in the But-For World Is Without Merit

39. In Part VI.C.4 of the Burtis Report, Dr. Burtis claims that Google would undertake the unprecedented step of imposing a charge on previously free Apps, making them no longer free to consumers.[77] The claim that Google would be willing to subject millions of consumers to a fee simply for downloading everything from Uber to Door Dash in a more competitive but-for world is without merit. As an initial matter, Dr. Burtis ignores that Google would continue to be compensated for any In-App Aftermarket services that it provides in a more competitive but-for world—the difference being that Google would have to compete on the merits to win the business of those developers by offering a more competitive take rate.

40. In addition, as the distributor of the initial application and the owner of the Android operating system, Google would have an incumbency advantage, providing it with a continued economic incentive to distribute as many applications as possible.[78]

41. The economics of two-sided platforms also makes it unlikely that Google would charge consumers to download previously free Apps. As explained in the Singer Report, because a large user base is critical to unlocking indirect network effects, two-sided platforms such as the Play Store are incentivized to provide free access to users (or even to subsidize access to users) to maximize participation by both users and developers.[79] This allows two-sided digital platforms to get "both sides on board."[80] Encouraging use of the platform by one group serves to attract the group on the opposite side of the platform.[81] In the instant case, allowing consumers to browse the Play Store for free and download free Apps creates a benefit for Google due to the indirect network effects

---

76. [redacted] See GOOG-PLAY-003605103 at Ex. 10 ([redacted] contract setting Direct Carrier Billing rate at [redacted] of Google's take rate starting July 2015); [redacted] GOOG-PLAY-001507837 at Ex. B ([redacted]); [redacted]

77. Burtis Report ¶¶186-189.

78. Even in the presence of substantial competition, I assume conservatively that Google's incumbency advantage would have allowed it to retain a substantial market share of 60 percent. Singer Report ¶192; ¶219.

79. Singer Report ¶¶270-271.

80. *Id.* ¶¶270-271.

81. Jean-Charles Rochet & Jean Tirole, *Platform Competition in Two-Sided Markets,* 1(4) JOURNAL OF THE EUROPEAN ECONOMIC ASSOCIATION 990, 991 (2003).

in attracting more developers and additional money Google can make by attracting advertisers.[82] As a consequence, the Play Store earns billions of dollars annually on ads that appear in the Play Store, with almost all of those revenues falling to the bottom line.[83] Free Apps in the Play Store are economically analogous to consumer inducements on a range of other two-sided platforms, such as credit cards (which often subsidize customers through loyalty programs), broadcast radio and television, (which offer free content to viewers or listeners), restaurant reservation software such as OpenTable (free to diners), and so on.[84]

42. According to a 2020 Google analysis, advertising in the Play Store "[redacted]"[85] As the analysis explains: "We already found our new business model ([redacted]), launched 4 years ago which means if we were to [redacted] we would still be overall ahead versus the pre-2015 baseline trajectory of our original business model."[86] This analysis is confirmed by Google's financials. For example, the Play Store generated [redacted] in high-margin advertising revenue alone in 2021.[87] Put differently, record evidence indicates that it would be economically irrational for Google to impose a fee on users to access free Apps and potentially forgo billions of dollars in annual revenue generated by its highly lucrative advertising business in the Play Store, which is only monetized by offering large volumes of users to advertisers and developers.

43. Deposition testimony of Jamie Rosenberg, Vice President of Strategy and Operations for Google's Platforms and Ecosystems Division, confirms that Google recognizes the benefits to the Android ecosystem of making the Play Store available to developers who offer their Apps *free of charge*, explaining that "supporting the broadest possible collection of apps…is good for Android."[88] Record evidence indicates that Google initially invested in and developed the Play Store

---

82. These benefits of driving consumer traffic through the Play Store likely explain, at least in part, why Google was willing to consider offering exceptionally low take rates to select developers such as [redacted]. For example, Google's notes from negotiations with [redacted] over [redacted], indicate that Google was open to not charging [redacted] take rate. *See* GOOG-PLAY4-007109523 at GOOG-PLAY4-007109526. That Google was willing to [redacted] to keep it on the platform illustrates the network effects that exist in this two-sided platform. By the same reasoning, Google would likely not be willing to lose a large number of applications (or users) by instituting an unprecedented charge for previously free Apps.

83. Singer Report ¶69.

84. *See, e.g.,* David Evans, *Two-Sided Market Definition* in MARKET DEFINITION IN ANTITRUST: THEORY AND CASE STUDIES (ABA Section of Antitrust Law) 1-35, 5 (2009), *available at* papers.ssrn.com/sol3/papers.cfm?abstract_id=1396751

85. GOOG-PLAY-006990552 at GOOG-PLAY-006990554.

86. GOOG-PLAY-006990552.

87. Singer Report ¶69.

88. Deposition of Jamie Rosenberg (Feb. 10, 2022) at 412:5-413:14 ("Q. And how does making the platform available to even developers who offer their apps for free help Google? A. Well, making the platform available to developers who offer the apps for free makes Android more successful as an operating system. It provides more

at a time when Apps and purchases of In-App Content generated relatively little revenue, and Google saw the value in the Play Store in terms of its contribution to the Android ecosystem.[89]

44. Thus, Google's network-driven incentive to draw as many users as possible into the Google ecosystem (including the GMS suite) would remain in a more competitive but-for world with substantially lower take rates. Google would continue to benefit from indirect network effects even under competition—the more consumers it retains (by eschewing an upfront user fee), the more Apps it can attract, and the more advertisers will come to its platform. Setting aside that Google would continue earning revenues from a more competitive take rate—and one several percentage points above the competitive rate[90]—any reduction in revenue from a lower take rate would be mitigated, at least in part if not wholly, by the continued advertising revenue generated from maintaining its user base. Google also would continue to bring users to its products, just as it does for Apps in the GMS suite that are free to consumers. Accordingly, Google likely would not impose even a modest fee on consumers for downloads of free Apps or a download fee on developers of free Apps that might be passed on to consumers.

45. Dr. Burtis ignores that when the Play Store cut its take rate from 30 percent to 15 percent for subscriptions longer than a year, it did not seek to offset the lost take-rate revenue by introducing new fees for free Apps. Similarly, Google did not announce any new fees for free Apps when it cut the take rate from 30 percent to 15 percent for the first $1 million in developer revenue. This suggests that, in a competitive but-for world in which Google would have to lower its take rate due to competition, it would still not seek to offset the lost revenue by charging consumers for downloading previously free Apps.

5. Dr. Burtis's Claim That Some Consumers Would Incur Higher Security Costs in the But-For World Is Without Merit

46. In Part VI.C.5, Dr. Burtis claims that some consumers would have incurred higher security costs in the but-for world. This claim is without merit. Dr. Burtis has provided no evidence or analysis to suggest that Google would stop investing in security in a competitive but-for world. Indeed, Google asserts that providing strong security is essential to integrity of Android and the Play

---

functionality to end users of an Android device. Android is important to Google. We want to see Android be successful. So supporting the broadest possible collection of apps we think is good for Android. Q. And what benefits does Google get from the success of Android? A. The success of -- well, Android brings -- brings a smart device operating system to billions of devices around the world. To the extent those devices are connected to the internet and able to do internet things, that's good for Google. More people engaging with Google over the internet is good for our desire to grow our services and make them successful and helpful. Q. And those services include the services that Google provides with respect to search, correct? A. Search, e-mail, maps, YouTube, you know, many things at Google depend on users having a device that is connected to the internet and performs well. Q. And many of those things are revenue generating for Google, such as search and advertising, right? Some of them are, yes.").

89. GOOG-PLAY-001501104 at GOOG-PLAY-001501105 (November 2009 email thread: "Google operates Android Market as a revenue-neutral service – we do not seek to profit off of application sales, and we invest in Market because it is essential to the open ecosystem"). The Play Store's financials show revenue of $51 million in 2011 and $204 million in 2012, with operating losses in both years. *See* GOOG-PLAY-000416245.

90. My economic model allows for Google to charge a markup over the competitive rate, even in the competitive but-for world, due to its incumbency advantage. Singer Report ¶220.

Store;[91] taking that statement at face value, there is no reason to suspect that Google would abandon its efforts to provide security in a competitive world.

47. Dr. Burtis also fails to recognize that monetization from the Play Store—and thus compensation for these basic features—does not end with the lifting of the Aftermarket Restrictions. Google would continue to capture commission-based revenue from the Play Store and from In-App Content sold through Google Play Billing in the competitive but-for world (again, at a premium above the competitive payment processing rate given its incumbency advantage). Although Google's take rate and market share in the but-for world would be lower than in the actual world, the size of the market would be substantially larger, given the substantial reduction in prices.[92] And Google would also continue to capture high-margin advertising revenue from the Play Store. These revenue streams would more than cover the cost of providing security, privacy, convenience, developer tools, and payment processing, while still allowing Google to earn a profit.[93]

48. Moreover, if the Aftermarket Restrictions were really necessary for delivering security, then one would expect Google to impose the Aftermarket Restrictions on all transactions. It does not; in fact, Google allows Apps to choose a third-party payment processor for certain types of purchases. Although Google requires the use of Google Play Billing for purchases of digital items, it *prohibits* the use of Google Play Billing for "purchases or rentals of physical goods," "purchases of physical services," and the "payment of a credit card or utility bill."[94] In addition, if the Aftermarket Restrictions were really necessary for delivering security, one would also expect to see similar restrictions employed by other app stores. As noted by Jared Sine, Chief Legal Officer of Match Group, in his April 2021 Congressional testimony: "If mandatory in-app payment was truly about security, it would stand to reason that every business would need to use it."[95] That competitive app store providers such as ONE Store and Aptoide do not impose aftermarket restrictions in their own stores, and that Google does not impose them on all types of purchases, implies that the Aftermarket Restrictions are not essential to preserving security in Android App Distribution Market.

49. In fact, the elimination of the Challenged Conduct in the Android App Distribution Market would invite more app store entry, and thus more competition on both price and non-price

---

91. *See, e.g., Secure an Android Device*, *available at* https://source.android.com/security/overview ("Android incorporates industry-leading security features and works with developers and device implementers to keep the Android platform and ecosystem safe. A robust security model is essential to enable a vigorous ecosystem of apps and devices built on and around the Android platform and supported by cloud services.").

92. *See, e.g.,* N. GREGORY MANKIW, PRINCIPLES OF MICROECONOMICS 67 (Cengage Learning 8th ed. 2018) [hereafter MANKIW]. ("This relationship between price and quantity demanded is true for most goods in the economy and, in fact, is so pervasive that economists call it the **law of demand**: Other things being equal, when the price of a good rises, the quantity demanded of the good falls, and when the price falls, the quantity demanded rises.") (emphasis in original).

93. In a more competitive but-for world, I estimate that the Play Store's but-for operating profit over the period 2015 – 2021 would come to $24.1 billion, compared with $41.5 billion in the actual world. *See* backup materials for this report.

94. *See Understanding Google Play's Payments Policy, available at:* https://support.google.com/googleplay/android-developer/answer/10281818.

95. Testimony for the U.S. Senate Judiciary Committee, Subcommittee on Competition Policy, Antitrust, and Consumer Rights, Jared Sine, Chief Legal Officer, Match Group, April 21, 2021 at 3.

dimensions such as security. Dr. Burtis's security claim is akin to claiming that Google would slow downloads in a competitive environment; if it tried, customer defection would be even greater. Consumers would benefit significantly from this added platform competition via multi-homing and steering, and Google would be punished if it were to unilaterally degrade the quality of its app store by paring back any of these features. If users value security as much as Google claims, then Google would retain strong economic incentives in a more competitive but-for world to invest in security to attract users.

D. Dr. Burtis's Speculation About Google's But For "Monetization Strategy" Fails to Demonstrate That Some Developers Would Have Been Worse Off in a More Competitive But-For World

50. In Part VI.D of the Burtis Report, Dr. Burtis discusses various purported "monetization strategies" that she claims Google would adopt in the but-for world. I note that many of Dr. Burtis's claims in Part VI.D of her report have already been addressed in Part I.C.4 above. In what follows, I focus on new claims. Importantly, none of Dr. Burtis's purported alternative monetization strategies has actually been adopted by Google. In addition, Google would have remained profitable in the but-for world even in the absence of alternative monetization strategies.[96]

51. Dr. Burtis also ignores that the Challenged Conduct confers substantially more market power on the Play Store than it would have enjoyed in a more competitive but-for world. The robust competition that the Play Store would face in the but-for world makes it even *less* likely that Google would have the incentive and the ability to implement the alternative monetization strategies set forth by Dr. Burtis, all of which would render the Play Store less attractive to consumers and developers.

1. Dr. Burtis's Claims Regarding a "Fee Per Download" Do Not Refute Classwide Impact

52. In Part VI.D.1 of the Burtis Report, Dr. Burtis cites to documents indicating that Google considered charging a fee each time an App is downloaded (or opened). As explained in Part I.C.4 above, imposing a fee per download would run counter to Google's economic interests as a two-sided platform and could jeopardize billions of dollars of Play Store profit, including high-margin advertising revenue, by breaking the virtuous cycle of indirect network effects. One Google document suggests that making "everything pay to play (charge for install or charge for update" would mean that Google"[w]ould lose ~75% of the apps on Play."[97]

53. Dr. Burtis offers the example of [redacted], which offers an App with over [redacted] downloads and total consumer spend of just [redacted].[98] As Dr. Burtis notes, this developer's 30 percent take rate comes to [redacted] (equal to 30 percent of [redacted]). If Google were to [redacted], then [redacted]

---

96. *See* n. 93, *supra*.

97. GOOG-PLAY-007346993 at -034. Another Google document suggests that Google's CEO in fact rejected early plans for an alternative business model charging by install. GOOG-PLAY-007887261 at GOOG-PLAY-007887269 ("Sundar ruled out $/install. Difficult to price install, opens, - shifts agitation elsewhere.").

98. Burtis Report ¶199.

would incur approximately [redacted] in fees.[99] Of course, [redacted] might not be willing or able to absorb [redacted] in fees for an App that generates revenue of approximately [redacted] of that amount ([redacted]). If [redacted] were to depart the Android platform or go out of business altogether, this would deprive Google of the opportunity of earning further revenue from [redacted]. In summary, this example merely illustrates why a "fee per download" is unlikely to be in Google's economic interest.

2. Dr. Burtis's Claims Regarding App or Developer Listing Fees Do Not Refute Classwide Impact

54. In Part VI.D.2 of the Burtis Report, Dr. Burtis suggests that in a more competitive world, Google might attempt to [redacted] and that this would effectively increase the take rate above 30 percent for the majority of developers.[100] Again, this alternative pricing strategy is purely speculative. Google currently charges a *one-time* fee of a modest $25 to open a developer account, presumably because that strategy is profit-maximizing; if [redacted] were profitable, Google would have [redacted] already. By Dr. Burtis's own calculations, [redacted] would impose higher costs on [redacted] of developers.[101] To the extent that these increased costs are economically significant, they could induce some developers to leave the platform entirely, or to purchase less advertising, or to scale back their offerings—all of which would undermine the virtuous indirect network effects that benefit the Play Store. To the extent that the increased costs from [redacted] are not economically significant (they would not be for larger developers), then Dr. Burtis's speculation is irrelevant. Moreover, Dr. Burtis implausibly posits that Google would charge a *higher* effective take rate—in excess of 30 percent—to the majority of developers in a *more* competitive but-for world.[102] Dr. Burtis ignores the elementary economic principle that increased competition would lower prices, not increase them.

3. Dr. Burtis's Claims Regarding Tiered Take Rate Structures Do Not Refute Classwide Impact

55. In Part VI.D.3 of the Burtis Report, Dr. Burtis speculates that some developers would not have been impacted "[i]f Google implemented a [redacted] in the but-for world[.]"[103] Dr. Burtis relies on evidence that Google considered "having a 30% rate for transactions associated with an individual consumer [redacted], but having a [redacted] rate apply [redacted]"[104] In Dr. Burtis's view, developers with Apps that generate transactions [redacted]

---

99. *Id.* As Dr. Burtis observes, even Google's (abandoned) proposal to [redacted] recognized the need to protect small developers, by proposing [redacted] to developers after the first 500,000 downloads. Burtis Report ¶199, n. 225; *see also* GOOG-PLAY-000336574 at GOOG-PLAY-000336588; GOOG-PLAY-006990552 at GOOG-PLAY-006990554, GOOG-PLAY-006990565.

100. Burtis Report ¶¶201-202

101. According to Dr. Burtis: "If, in the but-for world, Google Play charged these developers a [redacted] instead of the 30% service fee rate, more than [redacted] of these developers' fees would have been lower in the actual world than in the but-for world." *Id.* ¶202; Figure 9 (showing that over 60 percent of "Putative Developer Class Members" paid "Annual Service Fees of less than $99 during August 2016 – December 2021.").

102. *Id.* ¶202.

103. *Id.* ¶203.

104. *Id.* ¶204.

[redacted] would have paid a 30 percent take rate in both the actual world and the but-for world and therefore were not impacted.[105] Specifically, Dr. Burtis claims that about [redacted] percent of the proposed developer class have Apps [redacted] and therefore would have been no better off in a more competitive but-for world.[106]

56. Dr. Burtis is wrong. As before, Dr. Burtis's speculative alternative monetization strategy ignores that the [redacted] was never adopted by Google in the actual world in which the Play Store enjoys substantial market power owing to the Challenged Conduct. The [redacted] that Dr. Burtis contemplates would allow Google to extend its power into the In-App Aftermarket, presumably via the assistance of the Aftermarket Restrictions (or something similar), in violation of the counterfactual assumption an economist makes when modeling the but-for world. Put differently, Dr. Burtis's speculative [redacted] incorrectly presupposes that, in the but-for world, Google could tie Google Play Billing with App distribution. Moreover, because Google would face increased competition in the but-for world, Dr. Burtis has no basis to assume that Google would be able to profitably leave the take rate unchanged at 30 percent for [redacted] percent of developers. Dr. Burtis ignores that competition would impair Google's ability to implement a fee structure that would force developers to pay supracompetitive prices; if Google implemented a supracompetitive fee structure, developers and their customers would gravitate to a lower-cost platform.

4. Dr. Burtis's Claims Regarding Increases in the Play Store "Add-Load" Do Not Refute Classwide Impact

57. In Part VI.D.4 of the Burtis Report, Dr. Burtis cites evidence that Google considered "lowering service fee rates but [redacted]."[107] In other words, Dr. Burtis speculates that Google could have [redacted][108] Dr. Burtis argues that, although this would have benefitted "developers that value and are willing to pay for user acquisition and re-engagement,"[109] it could have harmed "developers that rely on [redacted][111]

58. Dr. Burtis provides no specific examples of developers (let alone consumers) that would have been harmed by this alternative monetization strategy. Such proof would require demonstrating that, in a more competitive but-for world, the benefits to developers of a substantially and permanently lower take rate would have been more than eliminated by [redacted]

105. *Id.* ¶205.
106. *Id.*; Table 7.
107. *Id.* ¶207.
108. *Id.*
109. *Id.* ¶208.
110. *Id.* ¶209.
111. For example, if Google were to sell more [redacted] this could crowd out other Apps that otherwise would have been displayed there.

[redacted] In a more competitive but-for world, Google likely would have lacked the ability and incentive to force such an offer on developers.

59. Dr. Burtis also speculates that larger developers may have been willing to [redacted] under this alternative monetization strategy.[112] But this does not imply that these developers (let alone their customers) would be harmed under this hypothetical monetization strategy; according to the record evidence she cites, these "dev[eloper]s value UA [User Acquisition] and re-engagement and are willing to pay for it."[113] Developers whose willingness to pay for a new feature exceeds its price would, by definition, achieve newfound economic gains.

60. As before, Dr. Burtis assumes nonsensically that, in a more competitive but-for world, Google would have had the incentive and the ability to implement alternative monetization strategies that Google declined to implement in the actual world.

E. Dr. Burtis's Additional Reasons That Developers Would Have Been Worse Off in a More Competitive But-For World Do Not Refute Classwide Impact

61. In Part VI.E of the Burtis Report, Dr. Burtis again offers a parade of horribles that would ensue in a more competitive but-for world. Dr. Burtis's claims contradict standard economics, which shows that competition benefits consumers. I address each claim below.

1. Dr. Burtis's Claim That Some Developers Would Have Incurred Higher App Distribution Costs Does Not Refute Classwide Impact

62. In Part VI.E.1 of the Burtis Report, Dr. Burtis claims that, in the competitive but-for world, developers would have had to offer Apps on more stores, thus incurring extra costs, and that these extra costs would have cancelled out the procompetitive benefits of substantially and permanently lower take rates. As an initial matter, a developer would not have to offer Apps on multiple stores in order to benefit from the results of competition in the but-for world. The mere threat of developers defecting to a competing platform, combined with actual defection (and steering) by other developers, would spur Google to decrease its take rate, in order to keep as many developers as possible on its platform.

63. Dr. Burtis's argument implies that higher output would represent an antitrust *harm*, a position inconsistent with standard economics, which posits the exact opposite—greater output benefits consumers.[114] To the extent developers would choose to offer their Apps on multiple platforms, they would attract more customers. Consumers who multi-home would be attracted by an App's availability on multiple platforms, thus potentially increasing the App's brand recognition and its reach. Dr. Burtis's claim implies nonsensically that developers would choose to offer multi-platform availability when the cost of doing so exceeds the benefits. Dr. Burtis provides no evidence that the incremental benefits of substantially and permanently lower take rates would be eliminated

---

112. Burtis Report ¶211.
113. *Id.*, n. 240, citing GOOG-PLAY-006990552 at -554 ([redacted]").
114. MANKIW, *supra*, at 301-302.

by the (presumably modest) incremental costs associated with operating on multiple app stores (all of which would be Android-compatible). Indeed, she relies on evidence indicating that "20 of top 100 apps on Play have published to [the] Amazon [appstore],"[115] despite the Challenged Conduct, and suggesting that more Apps would choose to publish on the Amazon Appstore if "Amazon gets more users."[116] In a more competitive but-for world, competing app stores would have expanded opportunities to attract a larger user base.

2. Dr. Burtis's Claim That Some Developers Likely Would Have Obtained Fewer App Distribution Services Does Not Refute Classwide Impact

64. In Part VI.E.2 of the Burtis Report, Dr. Burtis speculates that Google may have chosen to make fewer services available to some developers in a more competitive but-for world. This claim ignores that Google would likely have faced *greater* competitive pressure to expand its service offerings (or at least keep them the same) in order to compete with its rivals. Economic principles prescribe that as competition increases, firms compete more vigorously for customers on all dimensions, including services.[117] If Google attempted to degrade its service offerings in the but-for world, developers could switch to a competitive rival with similar offerings.

65. Dr. Burtis also speculates that developers that earn revenue from In-App Content could be worse off than developers that earn revenue from advertising, because Google "would have had economic incentives to invest more in tools for ads than in tools for IAPs."[118] As before, such speculation ignores the principle that competition likely would have compelled Google to *expand* its service offerings in the competitive but-for world. Google's internal documents suggest that Google was aware of the competitive pressure to expand its service offerings if it faced competition in the In-App Aftermarket. In a document planning an initiative in which "[redacted]" Google asked "[redacted]"[119] In that document, Google employees wrote that to achieve "Best in Class Platform Performance," it would need "[redacted]"[120]

66. Additionally, Dr. Burtis repeats her claim that "[d]evelopers that rely more on direct carrier billing [DCB] would have been worse off than developers not reliant on that payment form."[121] I have already responded to this claim in Section I.C.3 above.

---

115. Burtis Report ¶217, n. 243 (citing GOOG-PLAY-000560564 at 575 ("So far only 20 of top 100 apps on Play have published to Amazon. Many (like [redacted]) are concerned about the large and ongoing development / maintenance burden and potential loss of Go Global support. Decisions may change as Amazon gets more users.")

116. *Id.*

117. Department of Justice & Federal Trade Commission, *Horizontal Merger Guidelines* (08/19/2010) §1 ("For simplicity of exposition, these Guidelines generally discuss the analysis in terms of such price effects. Enhanced market power can also be manifested in non-price terms and conditions that adversely affect customers, including reduced product quality, reduced product variety, reduced service, or diminished innovation.")

118. Burtis Report ¶226.

119. GOOG-PLAY-007745829 at -829.

120. *Id.* at -831.

121. Burtis Report ¶227.

II. Dr. Burtis's Rebuttals To My Economic Analysis of Pass-Through and But-For Take Rates Are Without Merit

67. In this Section, I respond to Part VIII of the Burtis Report.

A. Dr. Burtis's Rebuttals to My Economic Analysis of Pass-Through Are Without Merit

68. In Part VIII.D of the Burtis Report,[122] Dr. Burtis argues incorrectly that my pass-through analysis "depends on unsupported assumptions, and includes no analysis of any service fee rate change or any cost change at all."[123] As detailed below, this is false: My pass-through analysis relies on standard economic methods showing how developers selling differentiated Apps would optimally adjust prices downward, according to the degree of concentration in their App category, in response to a decrease in marginal costs due to substantially and permanently lower take rates in the competitive but-for world. Dr. Burtis does not dispute that my pass-through calculations are based on standard economic models, nor does she dispute that I used standard econometric techniques to establish their applicability to the transactional data produced by Google. Instead, Dr. Burtis misleadingly criticizes the standard pass-through formulae derived from these standard economic models for their (relative) simplicity, ignoring all the calculations that work in the background to produce a deceptively straightforward result.

69. Dr. Burtis also claims incorrectly that my pass-through analysis "produces false results" based on pass-through analyses she performs purporting to demonstrate that various Apps' prices were unchanged after the developers experienced a decrease in the take rate.[124] As detailed in Part IV below, Dr. Burtis's pass-through analysis is fatally flawed. None of Dr. Burtis's pass-through analysis is capable of reliably measuring the effects of the substantially and permanently lower take rates for all or almost all developers that would have prevailed in a more competitive but-for world where Google's anticompetitive restraints were absent.

1. My Pass-Through Methodology Relies on Standard Economic Models That Fit the Data and Facts of This Case

70. This section responds to Part VIII.D.1 of the Burtis Report. In the Singer Report, I explained that all or almost all developers would pass through to consumers at least a portion of any savings from a substantially and permanently lower take rate in a more competitive but-for world.[125] This conclusion flows from the elementary economic principle that prices depend on costs and do not depend on the specific assumptions of any particular economic model.[126] To calculate the extent

---

122. Part VIII of the Burtis Report begins with VIII.D. (There are no sections titled VIII.A., VIII.B, or VIII.C).
123. Burtis Report ¶277.
124. *Id.*
125. Singer Report Part V.D.
126. *Id. See also* Jerry Hausman & Greg Leonard, *Efficiencies from the Consumer Viewpoint,* 17(3) George Mason Law Review 707, 708 (1999) ("What would be the effect on prices to consumers from the cost reduction? Economic theory makes a straightforward prediction: The decrease in cost will lead to a decrease in price, with the relationship between the decreases in cost and price depending on the shape of the demand curve.").

of pass-through, I used standard logit demand systems.[127] I did so only after confirming econometrically that the logit demand model fits the data well for the Play Store's various App categories: The price coefficients have the expected (negative) sign and are highly statistically significant; the logit demand model explains the vast majority (more than 85 percent) of the variation in the price of Apps and In-App Content.[128] My logit regressions control for extensive app-level variation, including over 200,000 App-level fixed effects.[129]

71. Dr. Burtis claims that my pass-through analysis "does not include service fees or prices,"[130] and that it "includes no analysis of any service fee rate change or any cost change[.]"[131] This is false. The pass-through rate calculation proceeds in two steps. Step one solves for the profit-maximizing *price* for a firm facing consumer demand characterized by the logit specification, taking into consideration the developers' marginal costs, including but not limited to *service fees*. In step two, the model solves for the pass-through rate, by calculating how much the profit-maximizing price will change in response to a change in marginal cost. It is therefore false for Dr. Burtis to claim that the logit demand model fails to account for developers' prices or marginal costs, both of which are incorporated in the analysis. My analysis relies on standard economic calculations showing how developers selling differentiated Apps would optimally adjust prices downward in response to a decrease in marginal costs brought about by substantially and permanently lower take rates in the competitive but-for world.[132]

72. Dr. Burtis faults the standard logit pass-through formula for its (relative) simplicity,[133] ignoring all the calculations of the optimal price and pass-through rate that underly

---

127. Dr. Burtis concedes that logit demand is "frequently used in economics[.]" Burtis Report ¶306. *See also* Singer Report ¶236, n. 507; Nathan Miller, Conor Ryan, Marc Remer, & Gloria Sheu*, Approximating the Price Effects of Mergers: Numerical Evidence and an Empirical Application* DOJ Economic Analysis Group Discussion Paper 12-8 (2012), at 2 (explaining logit is one of the demand systems "that commonly are employed in antitrust analysis.") Dr. Burtis cites to a very recent (2021) publication to suggest that some academics advocate alternatives to the logit model and other standard demand systems. Burtis Report ¶307, n. 365, (citing Steven Berry & Phillip Haile, "Foundations of Demand Estimation," *in* 4(1) *Handbook of Industrial Organization* (Elsevier 2021). But even that source explains that the logit demand model is a "common parametric demand specification," and reviews the logit demand model in detail. *Id.* at 8, and at 33-35. Moreover, the "random coefficient" demand estimation methods touted by academics such as Berry & Haile (2021), *supra*, as a possible alternative to standard demand systems such as logit, suffer from well-known computational problems, which can severely limit their applicability and accuracy when applied to real-world data sets. *See*, *e.g.* Christopher Knittel & Konstantinos Metaxoglou, *Estimation Of Random-Coefficient Demand Models: Two Empiricists' Perspective* 96(1) REVIEW OF ECONOMICS AND STATISTICS 34 (2014) ("We document the numerical challenges we experienced estimating random-coefficient demand models as in Berry, Levinsohn, and Pakes (1995) using two well-known data sets and a thorough optimization design. The optimization algorithms often converge at points where the first-and second-order optimality conditions fail. There are also cases of convergence at local optima. On convergence, the variation in the values of the parameter estimates translates into variation in the models' economic predictions. Price elasticities and changes in consumer and producer welfare following hypothetical merger exercises vary at least by a factor of 2 and up to a factor of 5.")

128. Singer Report ¶¶234-240. The only exception is the "Transportation" category, which accounts for less than [redacted] percent of consumer expenditures.

129. *Id*. Table 7.

130. Burtis Report ¶28.

131. *Id*. ¶277.

132. Singer Report ¶239, n. 516 (citing Nathan Miller, Marc Remer, & Gloria Sheu, *Using cost pass-through to calibrate demand*, 118 ECONOMICS LETTERS 451, 452-453 (2013)).

133. Burtis Report ¶280.

and produce a deceptively straightforward result. These calculations begin with the standard economic principle that firms set their prices to maximize profit—that is, by setting marginal revenue equal to marginal cost.[134] When marginal cost falls—due, in this case, to a substantial and permanent reduction in the take rate—developers will find that their prior prices are no longer profit-maximizing. Standard economics prescribes that developers will therefore decrease their prices until marginal revenue once again equals marginal cost.[135] When one solves for the extent of this price decrease in relation to the change in marginal costs, the result obtained from the logit demand model is that each developer will optimally decrease its price by an amount proportional to its share of revenues within the product category.[136]

73. This yields the economically intuitive result that developers with a lower market share (and thus less pricing power) will be inclined to pass through a larger proportion of a given cost decrease to consumers. For example, under the logit demand model, a developer with a market share of ten percent in a given product category has a pass-through rate of 90 percent: If marginal costs fall by $1 per transaction, the profit-maximizing response would be to lower its prices by $0.90. By aggregating across the market shares for all developers within a product category, I calculate the aggregate decrease in prices flowing from the decreased take rate that would have prevailed in the competitive but-for world.[137]

74. In faulting the standard logit pass-through formula for its (relative) simplicity, Dr. Burtis also ignores that there is nothing unusual in standard economics about pass-through calculations that boil down to (relatively) simple formulae. For example, as pointed out in the Singer Report, whenever the demand curve is assumed to be linear, the pass-through rate is always exactly 50 percent, regardless of how steep or flat the curve is.[138] For an assumed constant elasticity demand curve, the pass-through rate can also be calculated using a (relatively) simple formula.[139] As explained above, I selected the logit demand model because my econometric analysis confirms that it fits the data well.[140]

---

134. Miller, Remer, & Sheu, *supra*, at 452.

135. *Id.* at 452.

136. *Id.* at 453 (Equation 6 provides the formula for inverse of the pass-through rate, multiplied by negative one, which is: $-M/[M - Q_j]$, where M is the market size and Q_j is the quantity of product j. The pass-through rate is obtained by multiplying the expression by negative one and inverting it, which yields $[M - Q_j]/M$)). Dr. Burtis concedes that the logit pass-through rates are derived, not assumed. Burtis Report ¶28.

137. For example, Table 5 of the Singer Report shows that developer savings per transaction in the In-App Aftermarket would be $1.49. Because the pass-through rate is 89.9 percent, consumer prices would decline by $1.34 per transaction.

138. Singer Report ¶223.

139. Miller, Remer, & Sheu, *supra*, at 452-453 (showing that constant-elasticity or “log-linear” demand curves have a pass-through rate equal to $E/(E - 1)$, where E is the demand elasticity).

140. Singer Report ¶¶234-240 (standard logit regressions show that the price coefficients have the expected (negative) sign and are highly statistically significant; the logit demand model explains the vast majority (more than 85 percent) of the variation in the price of Apps and In-App Content). In any event, no matter what demand specification one chooses—logit, linear, constant-elasticity—it would follow that all or almost all consumers would have been better off in a competitive world. The elementary economic principle that prices depend on costs does not depend on the specific assumptions of any particular economic model. Singer Report Part V.D.

75. Dr. Burtis argues incorrectly that the Play Store categories employed in my pass-through analysis, which have been used consistently by the Play Store throughout the Class Period, "are not based on any economic analysis or reasoning[.]"[141] The Play Store's categories make economic sense because they reflect economically reasonable groupings of consumer tastes for different varieties of Apps, as recognized by a range of industry participants, including Google. According to Google, "Categories and tags help users to search for and discover the most relevant apps in the Play Store."[142] Record evidence indicates that Google uses App categories to conduct consumer research by application type. For example, a July 2017 consumer survey was designed "[t]o provide an understanding of how *within each category* users discover apps, the triggers and barriers that lead to a subscription."[143] A 2014 App Annie study commissioned by Google tracked download and revenue growth based on Google's categories.[144] An October 2015 Google study on differences in consumer spend between iOS and Android also used app categories to segment its analysis.[145] Google recognized in its 2016 Plan for "Google Play Developer Marketing" that "developers engage more with category-specific content" and planned for "category expertise, [to] drive actionable insights."[146]

76. The Play Store's categories are also used by industry analysts.[147] Developers, who presumably know their customers best, use Google's categories to sell their Apps in competition with other developers; they have clear incentives to select a meaningful category to maximize the value of their Apps.[148] If the developer of a "Parenting" App misclassified their App into the "Auto & Vehicles" category, that developer's ability to compete would likely be compromised. The evidence also shows that the Play Store's categories are economically meaningful to consumers, given their prominent display within the Play Store, and given that consumers can filter the Apps

---

141. Burtis Report ¶279.

142. Play Console Help, *Choose a category and tags for your app or game, available at* https://support.google.com/googleplay/android-developer/answer/9859673?hl=en#zippy=%2Capps%2Cgames (listing each of the Play Store's categories, with a description of each).

143. GOOG-PLAY-000294117.R at GOOG-PLAY-000294118.R (emphasis added). The survey, of users in Japan, targeted users of apps "in seven categories: Dating, Entertainment, Music, Health & Fitness, News, Education, Manga/Webtoon & Family."

144. GOOG-PLAY-000076773 at GOOG-PLAY-000076785 ("Although Games dominate revenue gains, growth is almost universal across categories."); GOOG-PLAY-000076766 (cover email noting that "we worked with App Annie on the attached report that was released this morning.").

145. GOOG-PLAY-000579868.R at GOOG-PLAY-000579870.R, GOOG-PLAY-000579878.R.

146. GOOG-PLAY-000303918.R at GOOG-PLAY-000303926.R, GOOG-PLAY-000303930.R.

147. *See, e.g.,* David Curry, *App Data Report,* BUSINESS OF APPS (2022) at 10 (chart showing Google Play categories by volume); *see also* SENSOR TOWER, *2021 – 2025 Mobile Market Forecast,* (2021) at 39, *available at* go.sensortower.com/rs/351-RWH-315/images/Sensor-Tower-2021-2025-Market-Forecast.pdf (showing projected consumer spending for top categories, including Games, Social, Entertainment, Comics, Productivity, and Health & Fitness); STATISTA, *Most popular Google Play app categories as of 1st quarter 2021, by share of available apps, available at* https://www.statista.com/statistics/279286/google-play-android-app-categories

148. Play Console Help, Choose a category and tags for your app or game, *supra* ("**Choose a category and tags for your app or game** You can choose a category and add tags to your app or game in Play Console. Categories and tags help users to search for and discover the most relevant apps in the Play Store. Users can view apps by using a browser and the Google Play app.") (emphasis in original).

displayed to them based on the Play Store categories.[149] Apple's App Store uses a similar set of categories, as seen below:

149. *See* Google Play Store, *Apps, available at* https://play.google.com/store/apps (click on drop-down menu).

TABLE 2: COMPARISON OF PLAY STORE AND APP STORE CATEGORIES

Play Store Category Name	App Store Category Name
Art and Design	Graphics and Design
Auto & Vehicles	N/A
Beauty	Lifestyle
Books & Reference	Books/Reference
Business	Business
Comics	Books
Communications	Social Networking
Dating	Social Networking
Education	Education
Entertainment	Entertainment
Events	Entertainment
Finance	Finance
Food and Drink	Food and Drink
Games	Games
Health and Fitness	Health and Fitness
House & Home	Lifestyle
Lifestyle	Lifestyle
Maps & Navigation	Navigation
Media & Video	Photo and Video
Medical	Medical
Music and Audio	Music
News and Magazines	Magazines and Newspapers/News
Parenting	Lifestyle
Personalization	Graphics and Design/Utilities
Photography	Photo and Video
Productivity	Productivity
Shopping	Shopping
Social	Social Networking
Sports	Sports
Tools	Utilities/Developer Tools
Transportation	Navigation/Travel
Travel & Local	Travel
Video Players & Editors	Photo and Video
Weather	Weather

Note: Separation of category name by "/" implies that the Play Store category could encompass multiple App Store categories.
Sources: https://developer.apple.com/app-store/categories/ ; Singer Report Table 7.

77. Although the logit demand system incorporates market shares, it bears emphasis that this need not be shares of a relevant antitrust product market. As DOJ economist Gregory Werden has observed, the market used in a logit demand system "may be more or less inclusive than a

relevant antitrust market."[150] The logit demand model also does not imply that all products in the market are interchangeable, but instead allows for product differentiation.[151] Products that are more attractive to most consumers (and thus have higher market shares) command more pricing power than less-attractive products with lower market shares.[152] What the logit demand system does imply is that developers in a given category pass through cost savings according to their dominance (or lack thereof) in the category, as measured by their market share within that category. For example, if Microsoft, which sells both Word and Excel, dominates the productivity category with its Microsoft 365 package (formerly known as Office, a bundle of Word, Excel, and PowerPoint), Microsoft is predicted to pass through a smaller portion of any cost reduction, all things equal. It is reasonable to assume that Microsoft 365 is a substitute for Google's bundle of productivity apps called Google Workspace. Both Microsoft 365 and Google Workspace are included in Google's "Productivity" category.

78. Although there is no requirement that the market share for the logit demand model be computed in a relevant antitrust market, it bears noting that antitrust has recognized "cluster markets," in which the market is comprised of items that are not always substitutes. As antitrust scholar Herbert Hovenkamp has noted, cluster markets have aggregated products as diverse as office supplies.[153] In a cluster market, a hypothetical monopolist over (say) paperclips, staples, paper, and other office supplies can profitably raise prices above competitive levels, given that many customers are likely to purchase many or all of their office supplies from the same source. By the same logic, a hypothetical monopolist over games ranging from "Thomas and Friends" to "Poker – Texas Hold'em" could also likely wield monopoly power, given that many households likely "need or at least prefer the convenience of"[154] purchasing games for all members of the family from the same source.

79. Even if one assumes that the "Games" category is overly broad, the same common methodology from the logit model can be used to calculate pass-through based on Google's seventeen subcategories of Games,[155] as Dr. Burtis concedes by performing the calculation

---

150. Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects* 70 ANTITRUST LAW JOURNAL 257 (2002). Economists make use of market shares in industries that may not perfectly correspond to an antitrust product market. *See, e.g.*, José Azar, Ioana Marinescu & Marshall Steinbaum, *Labor Market Concentration*, 57(3) JOURNAL OF HUMAN RESOURCES (2020) (finding that variation in wages could be explained by measures of labor market concentration within an occupational code using vacancy shares from CareerBuilder.com).

151. Gregory Werden & Luke Froeb, *The Effects of Mergers in Differentiated Products Industries: Logit Demand and Merger Policy* 10(2) JOURNAL OF LAW, ECONOMICS, & ORGANIZATION 407, 408 (1994) ("the logit model has direct policy relevance, since the 1992 Horizontal Merger Guidelines use it as the base case for the analysis of mergers in differentiated products industries.").

152. *Id.* at 410 (equation (3) shows that the own price elasticity for a given product (ε_j) decreases with the market share (π_j)).

153. Herbert Hovenkamp, *Digital Cluster Markets,* COLUMBIA BUSINESS LAW REVIEW at 31 (forthcoming 2022), *available at* https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3820062 ("In the *Staples* merger case, the court defined a cluster market…The expert concluded 'that a *monopoly* provider of consumable office supplies would charge significantly more to large customers than Staples and Office Depot today charge these same customers.'").

154. *Id.* at 7.

155. The seventeen subcategories are Action, Adventure, Arcade, Board, Card, Casino, Casual, Educational, Music, Puzzle, Racing, Role Playing, Simulation, Sports, Strategy, Trivia, and Word. *See* Google Play Store, Choose a category and tags for your app or game, *available at* https://support.google.com/googleplay/android-developer/answer/9859673?hl=en.

herself.[156] This approach is overly granular, as it assumes (for example) that action games are not substitutes for adventure games, that card games are not substitutes for board games, and so on. In any case, this approach would yield an overall weighted average pass-through rate of 77 percent across all categories, compared with an 89.9 percent pass-through rate in the Singer Report.[157] If this alternative pass-through rate of 77 percent is used, my economic models would continue to show classwide impact and damages, with only minor differences in the but-for take rates.[158]

2. Dr. Burtis's Pass-Through Analysis Is Incapable Of Reliably Measuring Pass-Through In A But-For World In Which All Or Almost All Developers Enjoy Permanently And Substantially Lower Take Rates

80. In Part VIII.D.1 of the Burtis Report, Dr. Burtis asserts that the pass-through rates from my standard economic models are incorrect based on her flawed "pass-through" analysis, to which I respond in Part IV below. Dr. Burtis also presents examples of Apps that charge the same price when purchased through the App website or through the Play Store.[159] As explained in Part I.C.1 above, this ignores the anti-steering restrictions that explicitly prevent developers from directing customers inside the Play Store to lower-cost options outside the Play Store. An App that charges the same price within the Play Store and on its website has simply not adopted a steering strategy in the actual world.

3. Dr. Burtis's Claim That My Pass-Through Analysis Is "Not Consistent With Economics" Is Without Merit

81. In Part VIII.D.3 of the Burtis Report, Dr. Burtis repeats the incorrect claims that my pass-through analysis does not account for marginal cost and that some (unspecified) developers might have zero marginal costs. I have responded to these claims in Parts I.B.1 and II.A.1 above. Dr. Burtis also argues that a calculation I provided to illustrate the economics of pass-through is inconsistent with the standard logit demand model that I employed to quantify pass-through in the but-for world.[160] Dr. Burtis is wrong. The calculation in question is based on the elementary economic principle that a profit-maximizing firm selling a single product will set its price based on the inverse of the elasticity of demand for that product.[161] That same principle of profit-maximization applies (albeit in a multi-product setting involving tens thousands of developers making

---

156. Burtis Report ¶312; Exhibit 55.

157. Within the Games category, the weighted average pass-through rate declines from 92 percent to 77 percent.

158. Using a 77 percent pass-through rate, the but-for take rate in the Android App Distribution Market is 23.5 percent, which differs by only one tenth of a percentage point from the but-for take rate of 23.4 percent calculated in Table 3 of the Singer Report. Using a 77 percent pass-through rate, the but-for take rate in the In-App Aftermarket is 14.5 percent, which differs by only three tenths of a percentage point from the but-for take rate of 14.8 percent reported in Table 5 of the Singer Report. Aggregate damages in the Android App Distribution Market using a 77 percent pass-through rate come to $15.2 million, compared with $18.8 million calculated in Table 3 of the Singer Report. Aggregate damages in the In-App Aftermarket using a 77 percent pass-through rate come to $4.03 billion, compared with $4.71 billion calculated in Table 5 of the Singer Report.

159. Burtis Report ¶297; Table 8.

160. Burtis Report ¶¶299-305.

161. Singer Report ¶224. *See also* William Landes & Richard Posner, *Market Power in Antitrust Cases,* 94(5) HARVARD LAW REVIEW 937 (1981).

interdependent pricing decisions) when the logit demand model is used to calculate the pass-through rate.[162]

4. Dr. Burtis's Claim That the Basis for My Pass-Through Analysis Is "Fundamentally Flawed" Is Without Merit

82. In Part VIII.D.4 of the Burtis Report, Dr. Burtis repeats meritless critiques of the standard economic methods I employed to estimate pass-through. I have already responded to these claims in Part II.A.1 above. In addition, Dr. Burtis claims that my overall pass-through rate "mask[s] variation."[163] Yet I calculate pass-through rates for each of the Play Store's categories, display them in the same table as the overall pass-through rate (Table 8 of the Singer Report), and use the category-by-category pass-through rates to perform illustrative damages calculations (Tables 13-14 of the Singer Report).

83. Dr. Burtis also argues that my category-level pass-through rates "mask variation" because the pass-through rates are a weighted average of the pass-through rates across all of the developers within a given category.[164] Indeed, Dr. Burtis asserts that any analysis would be unacceptable unless it took into account individual idiosyncrasies associated with each consumer, developer, App, and each item of In-App Content.[165] This ignores Google's highly uniform and formulaic take-rate structure built from a standard headline rate of 30 percent, as well as the Challenged Conduct, which imposes uniform restrictions on all developers, Apps, subscriptions, and In-app Content. All of this justifies a category-wide approach, including using a weighted average of the developer-specific pass-through rates within a given category from my standard logit demand model. It bears emphasis that Dr. Burtis concedes (for purposes of her analysis here) the relevant markets as they are defined in the Singer Report,[166] which further justifies a market-wide economic analysis when calculating but-for headline rates in the Android App Distribution Market and In-App Aftermarket. In any event, if the factfinder requires App-specific pass-through rates for the allocation of damages at the merits phase, I have provided a model that can perform that precise task.

5. Dr. Burtis's Claim That My Pass-Through Rates Are Unreliable Is Without Merit

84. In Part VIII.D.5 of the Burtis Report, Dr. Burtis repeats her claim that developers would avoid charging prices that do not end in "99" in the competitive but-for world.[167] I have responded to this claim in Part I.B.2 above. Dr. Burtis also asserts that the pass-through rates from the standard logit demand model are "not credible because, in some instances, the rates change month to month as shares change month to month."[168] Dr. Burtis is wrong. *First*, I am not offering an opinion on any developer's pass-through rate for a specific month (or week, or day). As explained

162. Miller, Remer, & Sheu, *supra*, at 452-453.
163. Burtis Report ¶312.
164. *Id.* ¶312.
165. *Id.* Part IV.A; ¶¶103-104; ¶307.
166. *Id.* ¶¶42-43.
167. Burtis Report ¶¶313-315.
168. *Id.* ¶316.

above, the but-for world is based on a long-run equilibrium of substantially and permanently lower take rates. *Second*, the examples presented in Figure 14 of the Burtis Report are cherry-picked. When I apply standard econometric methods to the entire data set, the results demonstrate that the month-specific pass-through rates deviate from the time-averaged pass-through rates by an immaterial 2.0 percentage points on average.[169]

6. Dr. Burtis Claims Incorrectly That Determining Pass-Through Rates Requires an Individualized Analysis

85. In Part VIII.D.6 of the Burtis Report, Dr. Burtis asserts that individualized analysis is necessary because my pass-through model yields different results when compared to Dr. Williams's pass-through model.[170] The question of whether my pass-through rate is consistent or inconsistent with Dr. Williams's analysis has no bearing on whether individualized analysis is required. In any case, as I explain in Part IV below, Dr. Williams's pass-through analysis is not reliable.

B. Dr. Burtis's Rebuttals to My Economic Analysis of But-For Take Rates Are Without Merit

86. In this section, I respond to Part VIII.E of the Burtis Report. Dr. Burtis argues incorrectly that I assume Google chooses a single uniform take rate without exception.[171] As I explained in Part II.A.1 above, although Google's take-rate structure is highly uniform and formulaic, my analysis does not assume a uniform take rate across all developers in a more competitive but-for world. Instead I use standard economic methods to solve for a competitive but-for *headline* rate, which tracks Google's highly uniform and formulaic take-rate structure in the actual world built around a headline rate of 30 percent. I have also shown that damages can be calculated for individual Class members using common methods, even after taking into account the limited number of developers who received discounts relative to Google's standard 30-percent take rate.[172]

87. Dr. Burtis suggests incorrectly that my two-sided market does not make prices on both sides of the platform "part of the same equilibrium decision."[173] In fact, prices on both sides of the market are taken into account in my models: In the Singer Report, I implemented two-sided market models in which the price on one side of the market is held constant (*not* ignored), while

---

169. This result is obtained by regressing the logit pass-through rates on App fixed effects. The standard deviation of the residuals (or the root mean squared error) of the regression is 0.020.

170. Burtis Report ¶¶317-322.

171. *Id.* ¶332.

172. Singer Report ¶177, n. 376; *see also* Singer Report Part VII.

173. Burtis Report ¶331, n. 396.

solving for the optimal price on the other side of the market.[174] In any case, classwide impact and damages can also be demonstrated by implementing models that simultaneously solve for optimal prices on both sides of the market, with results similar to those in the Singer Report.[175]

1. My Analysis of But-For Take Rates Uses Standard Economic Models and Does Not Assume Classwide Impact

88. In Part VIII.E.1 of the Burtis Report, Dr. Burtis claims that my analysis "treats all developers as if they are all the same."[176] This is false. As I explained in Part II.A.1 above (and immediately above), I do not assume that all developers would be subject to a uniform take rate in the but-for world. Nor do I assume that all developers would have had the same pass-through rate: As I explain in Parts I.B and II.A above, I use standard economic methods to estimate equilibrium pass-through rates for thousands of developers making thousands of interdependent decisions for each of their Apps, taking into account prices, marginal costs, and other developer-specific characteristics. Dr. Burtis also reiterates the incorrect claim that developers selling at a price of $0.99 would have been worse off in the but-for world.[177] I have responded to this claim in Part I.A.2 above.

2. Dr. Burtis's Critiques of the Inputs to My Standard Economic Models Are Without Merit

89. In Part VIII.E.2 of the Burtis Report, Dr. Burtis criticizes the standard economic models that I employ because they rely on aggregate inputs.[178] Dr. Burtis ignores that standard economic models commonly rely on aggregate inputs. For example, a textbook "supply and demand model" uses aggregate data on demand shifters (e.g. median income), and supply shifters (e.g., an index of the price of raw materials) to determine a market-clearing price that may apply to thousands (or millions) of heterogeneous consumers.[179] Dr. Burtis also ignores that aggregation is both pervasive and economically necessary in the markets at issue here: Google's highly uniform and formulaic take rate structure, based on a common 30 percent headline rate, is applied to tens of

---

174. In the Singer Report, equation (V.5), which forms the basis for the presentation of the two-sided market model in Section V.B and Appendix 4 of the Singer Report, gives a competitive equilibrium condition that includes both buyer-side prices (P_B) and the seller-side take rate (t), as well as both buyer- and seller-side elasticities. By way of pass-through, consumers are necessarily affected in these models by the adjustment of the take rate to a competitive level, a feature that my modeling accounts for. Equation (V.12), which forms the basis for the presentation of the two-sided market model in Section V.E, also gives a competitive equilibrium condition that includes both the buyer-side price (P_B) and a seller-side take rate (t), thereby taking both sides into account when solving for the but-for buyer-side subsidy.

175. When I solve simultaneously for optimal prices on both sides of the market, the model shows that the Play Points buyer-side subsidy would be expanded to $0.56 per transaction (approximately 6.4 percent of consumer spend), up from [redacted] per transaction in the actual world. The but-for take rate is 27.5 percent, down from [redacted] percent in the actual world. The combined effect of a higher buyer-side subsidy and a lower take rate yields aggregate damages to the Class of $2.63 billion. By way of comparison, if the take rate is held constant and the buyer-side subsidy is permitted to vary, aggregate damages are $2.71 billion. *See* Singer Report Table 12. If the buyer-side subsidy is held constant and the take rate is permitted to vary, aggregate damages are $2.35 billion. *Id.*

176. Burtis Report ¶337.

177. *Id.* ¶338.

178. *Id.* ¶¶339-342.

179. JEFFREY WOOLDRIDGE, INTRODUCTORY ECONOMETRICS: A MODERN APPROACH, 552-555 (THOMPSON 4TH ED. 2009).

thousands of heterogeneous developers. It would be economically inefficient for Google to negotiate individually with each of them, so Google sets a highly uniform market pricing structure instead.

90. In the Singer Report, I examined the market shares of historically dominant firms that went on to face some degree of competition in different industries.[180] Based on this evidence, I assumed conservatively that in a competitive but-for world Google would retain a substantial market share of 60 percent in the In-App Aftermarket and in the Android App Distribution Market, given its incumbency advantage and network effects. This was approximately AT&T's market share in the long-distance market after competitive entry, and it is substantially above the market share of Visa in e-commerce or of Alcoa in aluminum manufacturing.[181] Dr. Burtis argues incorrectly that there is no evidence that AT&T provides a valid competitive benchmark for Google.[182] Dr. Burtis is wrong. She ignores that AT&T was a classic network monopolist that leveraged monopoly power in the (ancillary) long-distance market from its monopoly in local service, before eventually being forced to open the long-distance market to competition—just as Google is a network monopolist that leveraged its power in the Android App Distribution Market into the In-App Aftermarket Services Market.[183]

91. Dr. Burtis also ignores that my selection of a 60 percent but-for market share was informed by market shares in other industries, including e-commerce.[184] Here I present evidence from additional industries confirming that my 60 percent estimate is, if anything, likely conservative. Netflix, which made streaming video on demand (SVOD) a staple of home entertainment, dominated the market for streaming video services for years.[185] As recently as 2014, approximately nine out of every ten SVOD households were Netflix subscribers.[186] More recently, Netflix's market share has eroded as competitors such as Amazon Prime, HBO Max, and others have gained at its expense.[187]

---

180. Singer Report ¶¶216-219.

181. Visa is an historically dominant global payments technology company. Alcoa is historically dominant in aluminum manufacturing. *See, e.g.,* STATISTA, Market share of global general purpose card brands American Express, Diners/Discover, JCB, Mastercard, UnionPay and Visa from 2014 to 2020, based on number of transactions, *available at* https://www.statista.com/statistics/278970/share-of-purchase-transactions-on-global-credit-cards/; *see also* Singer Report ¶¶216-219.

182. Burtis Report ¶345.

183. Dr. Burtis offers only a single argument as to why AT&T does not represent such a benchmark, claiming that AT&T's market share would have "likely varied across geography." Burtis Report ¶346, n. 411. Dr. Burtis offers no support for her claim, nor does she explain why it is relevant here.

184. Singer Report ¶¶216-219.

185. *See, e.g.,* James Brumley, *Netflix is Losing Market Share, but This is the Actual Risk to Shareholders*, THE MOTLEY FOOL, Apr. 25, 2021, *available at* https://www.fool.com/investing/2021/04/15/netflix-is-losing-market-share-but-thats-not-the-a/ ("Netflix (NFLX -1.73%) is losing market share to be sure -- but consider the circumstances. It was the first company to make streaming video a mainstream phenomenon, and for years, it was the only serious name in the business. It's only natural that the recent launches of big rival services such as Disney's (DIS 0.00%) Disney+ and AT&T's (T 1.73%) HBO Max would chip away at Netflix's share of the on-demand video space.").

186. Nielsen, *The Total Audience Report Q4 2014*, at 5, *available at* https://www nielsen.com/wp-content/uploads/sites/3/2019/04/total-audience-report-q4-2014.pdf (showing 40.3 percent of US TV households with SVOD, and 36 percent of US TV households subscribing to Netflix).

187. Georgina Tzanetos, *Netflix Loses 31% Market Share as Streaming Rivals Gain Loyal Subscribers*, YAHOO!, (April 7, 2021), *available at* https://www.yahoo.com/video/netflix-loses-31-market-share-204537722 html.

As seen below, as of Q4 2021, Netflix had a streaming share of just 25 percent, compared to Amazon Prime's 19 percent, Disney + and Hulu at 13 percent each, and HBO Max at 12 percent.[188]

92. The personal computer (PC) market has also seen dominant firms lose substantial share. The IBM brand was nearly synonymous with the industry for decades. However, competition from other PC makers such as Compaq and Apple Computer dissipated IBM's market share, which fell from 80 percent to 20 percent in the decade between 1982 and 1992.[189] In 2004, IBM sold its personal computer business to Lenovo, which maintained a 24.6 percent worldwide market share in 2021, compared to 21.1 percent for HP, 19.5 percent for Dell, and around 7 percent each for Apple, Acer, and ASUS.[190]

93. Competitive entry in the Internet browser market eroded the market share of Microsoft's Internet Explorer. In 2004, Internet Explorer enjoyed 95 percent market share.[191] By June 2010, its market share had slipped to 53.8 percent, as Firefox (30.6 percent), Apple Safari (6.8 percent), Google Chrome (5.7 percent) competed for users.[192] Recently, Google Chrome has supplanted Microsoft's browser offering (now called Edge) at the top of the market, with a usage share of 65 percent of all browsers, compared to 19 percent for Safari and only 4 percent for Edge.[193]

94. In summary, ample evidence from a range of industries, including network industries where market power was leveraged from the core into an ancillary market (AT&T and Microsoft), supports my conclusion that Google's share of payment processing would decrease substantially in the competitive but-for world. If anything, the evidence shows that 60 percent is likely a conservative estimate of Google's market share after competitive entry.

---

188. Joe Wituschek, *Apple TV+ gains market share in the United States while Netflix loses it*, JUSTWATCH, Jan. 24, 2022, *available at* https://www.imore.com/apple-tv-gains-market-share-united-states-while-netflix-loses-it. *See also* Brumley, *supra* ("Data from market intelligence outfit eMarketer lets us fleshes out this trend with some numbers. It reports that Netflix secured 36.2% of the U.S. over-the-top television industry's revenue in 2020, down from 44.4% in 2019. By 2022, its share is expected to be down to 28.4%, and almost even with Disney's slice of the U.S. streaming market.").

189. James W. Cortada, *How the IBM PC Won, Then Lost, the Personal Computer Market - Not even Big Blue could keep up with its creation's success*, IEEE SPECTRUM, July 21, 2021, *available at* https://spectrum.ieee.org/how-the-ibm-pc-won-then-lost-the-personal-computer-market.

190. Gartner Press Release, Gartner Says Worldwide PC Shipments Declined 5% in Fourth Quarter of 2021 but Grew Nearly 10% for the Year, Jan. 12, 2022, *available at* https://www.gartner.com/en/newsroom/press-releases/2022-01-12-gartner-says-worldwide-pc-shipments-declined-5-percent-in-fourth-quarter-of-2021-but-grew-nearly-10-percent-for-the-year.

191. TheCounter.com, Browser Stats, April 2004, *archived at* https://web.archive.org/web/20111101195133/http://www.thecounter.com/stats/2004/April/browser.php.

192. AT INTERNET INSTITUTE, Are we heading towards the end of Internet Explorer's reign in Europe?, July 27, 2010, *archived at* https://web.archive.org/web/20100806153329/http://www.atinternet-institute.com/en-us/browsers-barometer/browser-barometer-june-2010/index-1-2-3-205 html.

193. STATCOUNTER GLOBALSTATS, Browser Market Share Worldwide, October 2021, *available at* https://gs.statcounter.com/browser-market-share#monthly-202110-202110-bar.

3. My Standard Economic Models Show a Reasonable and Expected Relationship Between Competition and Lower Prices

95. In Part VIII.E.3 of the Burtis Report, Dr. Burtis critiques my standard economic models because they "find any change in competition leads to lower rates for all developers."[194] Yet the relationship between competition and prices is foundational to antitrust and economics. Dr. Burtis also claims that my models predict that "if, absent the alleged conduct, Google Play's market share fell by a single percentage point, the rate for all developers' IAPs would fall [redacted] [percentage] points (to [redacted])."[195] Dr. Burtis obtains this result by assuming counterfactually that the Play Store faces robust competition from rivals, such that the Play Store must drop its take rate substantially to prevent its rivals from expanding.[196] In the actual world, the Challenged Conduct precludes such competition. If Dr. Burtis's assumption is removed, reducing Google's market share by one percentage point causes the take rate to fall by just [redacted] percentage points.[197]

96. In this section, Dr. Burtis reiterates her incorrect argument that Google would reduce its take rate only in a targeted fashion in the face of robust and widespread competition in the but-for world.[198] I have responded to this claim in Part I.A.1 above.

4. Dr. Burtis's Speculations Regarding Alternative Monetization Strategies Are Not Appropriate To Include In My Economic Models

97. In Part VIII.E.4 of the Burtis Report, Dr. Burtis reiterates her incorrect claims regarding alternative monetization strategies. I have already responded to these claims in Part II.B.4 above.

III. Dr. Burtis's Critiques of My Analysis of Play Points Are Without Merit

98. In this Section, I respond to the arguments set forth in Part IX of the Burtis Report, in which Dr. Burtis criticizes an alternative model of impact and damages that I presented that works through direct consumer subsidies.[199] Dr. Burtis argues that I failed "to consider that Google's Play Points program provides benefits to a relatively small percentage of U.S. consumers,"[200] as "less than [redacted] of U.S. consumers participated in the program and only up to [redacted] of U.S. consumers earned and redeemed Play Points."[201] That "only" [redacted] of U.S. consumers participated in Play Points in the actual world—when the benefits of Play Points are comparatively meager—is hardly

194. Burtis Report ¶¶347-349.
195. *Id.* ¶¶348.
196. Specifically, Dr. Burtis sets the elasticity of supply of Google's would-be rivals to the same value used to model the competitive but-for world ($E_s = 4.38$). *See* Singer Report ¶216.
197. Specifically, if one sets the elasticity of rival supply to zero, ($E_s = 0$) and reduces the Play Store's market share by one percentage point, then the take rate falls by [redacted] percentage points in the In-App Aftermarket. *See* Singer Report Table 5. Dr. Burtis concedes that the Android App Distribution Market is not sensitive to her proposed adjustment; reducing Google's market share by one percentage point in the Android App Distribution Market lowers the take rate by just [redacted] percentage points, to [redacted] percent. Burtis Report ¶348.
198. Burtis Report ¶349.
199. Singer Report ¶¶245-256.
200. Burtis Report ¶358.
201. *Id.*

evidence that participation in a more competitive but-for world would not be substantially greater. In addition, that "only" about [redacted] percent[202] of those who participated earned and redeemed Play Points in the actual world does not prove that a more generous program would not see more widespread redemptions. Consumers would have enhanced economic incentives to enroll and participate in a Play Points offering more valuable incentives in the but-for world, just as consumers have more incentives to participate in a more generous credit card rewards program than a less generous one. In a more competitive but-for world, Google would also be incentivized to facilitate consumer participation in Play Points. For example, consumers could be automatically enrolled in Play Points. Discounts could be automatically redeemed at the point of purchase or even dispensed through a "cash-back" program.[203]

99. Dr. Burtis also assumes incorrectly that Play Points that are not redeemed have no value. This is incorrect for at least two reasons. *First*, Play Points that have not yet been redeemed today may still be redeemed in the future—in other words, they have an intrinsic option value. *Second*, that different consumers may value Play Points differently does not make them worthless. A $10 gift card for Chick-Fil-A may be worth considerably more to some than others, just a jar of change accumulating in the closet might be worth more to some consumers than others. That different consumers may place different values on points, gift cards, or even money does not imply that they lack economic value.

IV. NEITHER DR. BURTIS NOR DR. WILLIAMS PROVIDES A RELIABLE ANALYSIS OF PASS-THROUGH IN A MORE COMPETITIVE BUT-FOR WORLD

100. Drs. Burtis and Williams both claim to have implemented pass-through analyses that purport to estimate the change in the prices that developers would charge consumers for Apps or In-App Content as a result of a change in developers' costs arising from a change in Google's take rate.[204] As detailed below, neither Dr. Burtis nor Dr. Williams provides a pass-through analysis capable of reliably measuring the effects of the substantially and permanently lower take rates for all or almost all developers that would prevail in a more competitive but-for world where Google's anticompetitive restraints were absent. Both pass-through analyses are unreliable, biased, and virtually guaranteed to underestimate pass-through in a competitive but-for world that offers developers an opportunity to steer customers to lower-cost app stores or payment processors. No "corrections" to their methods could produce reliable pass-through estimates, given that no data exist to directly measure pass-through in the absence of the Challenged Conduct, which still remains in place.

101. Real-world examples of pass-through also undermine the pass-through analysis of Drs. Burtis and Williams. Certain developers such as Netflix, Spotify, and Tinder have been able to avoid Google's In-App Aftermarket Restrictions and to steer customers towards their website by offering a lower price; other developers (including Google's YouTube) have adopted

---

202. Equal to [[redacted] percent]/[[redacted] percent].

203. *See, e.g.,* Discover, Discover it Cash Back Card, *available at* https://www.discover.com/credit-cards/cash-back/it-card html

204. Burtis Report ¶¶173-178; ¶¶280-295; Table 5, Figure 13, Exhibits 35-36, Exhibits 48- 52; Williams Report ¶¶76-88; ¶¶111-124.

a similar strategy with respect to Apple's App Store. The implied pass-through rates from these real-world examples of pass-through are far above the estimates of Drs. Burtis and Williams, even when pass-through is calculated conservatively.[205]

A. Review of Dr. Burtis's and Dr. Williams's Pass-Through Analyses

102. Below I briefly review the pass-through analyses presented by Drs. Burtis and Williams. I also highlight significant flaws in their analysis that are evident from this review, and which preview some of my broader critiques in later sections.

1. Dr. Burtis's Pass-Through Analysis

103. Dr. Burtis's pass-through analysis is based on a comparison of prices for three groups of developers, grounded in three episodes in which take rates were modified: (1) the Play Store's 2018 take-rate reduction for subscription developers; (2) the Play Store's mid-2021 take-rate reduction for the first $1 million in developer revenue; and (3) take-rate reductions for special programs such as LRAP, which granted take rate discounts to a limited number of high-profile developers such as [redacted] that Google was attempting to attract for "living room" based-Android devices.[206] For each of these groups, Dr. Burtis compares the price of the App (or In-App Content) one month before and one month after the take-rate decrease, and concludes there is evidence of pass-through only if the latter price is lower than the former price.[207] Dr. Burtis also compares prices six months before and six months after the take rate decrease.[208] Dr. Burtis's pass-through analysis therefore ignores the long-run effects of lower take rates by construction.[209] Dr. Burtis provides no economic basis for restricting her analysis to one month (or six). These time horizons are brief, compared to a more competitive but-for world in which the Challenged Conduct is absent for all relevant time periods. This error is compounded by the fact that a decrease in the take rate does not immediately flow through to a developer's financials. For example, if a developer's take rate falls from 30 percent to 15 percent in a given month, the developer's average take rate over the prior year is approximately 29 percent.[210]

104. Dr. Burtis offers three cherry-picked examples (out of tens of thousands of developers) involving time horizons of more than six months.[211] In 2018, the Play Store decreased

---

205. Singer Class Report ¶¶242-243; Table 9 (showing real-world pass-through rates ranging from 33 to 105 percent).

206. Burtis Report ¶174. Only one tenth of one percent of the proposed developer class qualified for any of these special programs. *See* Part I.A.1, *supra.*

207. *Id.* ¶176; Table 5; Exhibit 36. When comparing the price of In-App purchases, Dr. Burtis uses a time horizon of four to eight months. *See* Burtis Report Exhibit 36, n. [2] ("Price changes for IAPs are identified by comparing prices in June 2021 to average prices in October 2021 and February 2022. Price changes for 'Scraped Data' SKUs are identified by comparing the retail prices in the scraped data in the month before and the month after the rate change occurred. Price changes for all other purchase types are identified by comparing weighted average net prices in the month before the rate change occurred to the month after.")

208. *Id.* ¶293; Exhibit 50.

209. Despite this limitation, Dr. Burtis finds that approximately 22 percent of initial downloads saw their prices decrease within six months of a decrease in the take rate. *See* Burtis Report Exhibit 50.

210. Note that [0.3 x 11 + 0.15]/12 = 0.2875.

211. Burtis Report ¶¶280-288 (showing subscription prices and take rates for [redacted]).

take rates for developers offering subscription services in which the subscriber maintained their subscription for at least twelve continuous months.[212] Unlike the vast majority of subscription products, the take rates for these three cherry-picked examples ([redacted]) dropped from 30 percent to 15 percent shortly after January 2018 because the products are annual renewals for *existing* subscribers only.[213]

105. Searching for pass-through among these specialized products makes little economic sense, because it runs contrary to the elementary economics of subscription pricing, which incentivizes "front-end" discounts to *new* customers.[214] Pass-through cannot be reliably measured by searching for "back-end" discounts offered to subscription developers' *least* price-sensitive consumers (renewals for existing customers).[215]

106. In addition, even if developers had an economic incentive to offer discounts to their least price sensitive customers, Google's developer pricing policies would have made doing so difficult (if not impossible) for most developers. As detailed below, the Play Store's developer pricing interface made it difficult or impossible for most developers even to implement "back-end" discounts for existing customers.[216]

107. Moreover, the data show low and declining transaction volume for Dr. Burtis's cherry-picked products, indicating they are not representative of stable product offerings that would drive developer pricing decisions in a more competitive but-for world.[217]

2. Dr. Williams's Pass-Through Analysis

108. Dr. Williams's pass-through methodology hinges on what he claims is a "natural experiment" created by Google's decision to lower the take rate from developers with subscription Apps from 30 to 15 percent beginning in January 2018—with the 15 percent take rate applied only

---

212. *Id.* ¶174, n. 193 ("In 2018, Google reduced the service fee rate on subscription IAPs when the consumer's subscription extended beyond a year.")

213. *Id.* Figures 10-12; ¶282, n. 337 ("When Google initiated the program and throughout the period [redacted] sold subscriptions in its [redacted] app, consumer subscriptions apparently are annual renewals. Thus, its service fee rate dropped from 30% to 15% immediately after Google made the service fee rate reduction and continued to be 15% throughout the time it sold subscriptions.").

214. *See* Part IV.D below.

215. *See* Part IV.D below.

216. *See* Part IV.E below.

217. For example, the cherry-picked [redacted] subscription product had fewer than ten transactions per month after mid-2018. By mid-2019, there were fewer than five transactions per month. There were no transactions at all for most of 2020, and none after October 2020. *See* Appendix Figures A1-A3.

to customers with a subscription in place for at least twelve continuous months.[218] Based on his "natural experiment" comparing prices in a "control group" to prices in a "treatment group" using a difference-in-differences ("DID") regression, Dr. Williams concludes that "the overall pass-through rate is minimal, no greater than 7.3%."[219] Dr. Williams implements a difference-in-difference ("DID") regression that attempts to measure the decrease in the average price charged by developers in a "treatment group," relative to a "control group."[220] Dr. Williams claims his "treatment group" consists of "subscription products" that "received the 15% service fee after January 1, 2018."[221] Dr. Williams asserts his "control group" consists of "subscription products" whose take rates remained fixed at 30 percent.[222]

109. Dr. Williams's DID regression purports to measure the extent to which monthly prices in the "treatment group" diverged from monthly prices in the "control group" after the start of the "treatment period," defined as the 42 months from January 2018 through June 2021.[223] The DID regression includes fixed effects for each "product."[224] Accordingly, Dr. Williams's DID regression measures pass-through only to the extent that prices of the "products" in the "treatment group" decline during the "treatment period," relative to prices of the "products" in the "control group."[225]

110. To obtain what he terms "clean effects of the treatment," Dr. Williams discards all "products" from his "treatment group" that do not have take rates of exactly 15 percent after 2017.[226] Similarly, he discards all "products" in the "control group" that do not have take rates of exactly 30 percent.[227] Dr. Williams's "treatment group" accounts for just 1.5 percent of the revenue of the developers he purports to analyze during his "treatment period." Dr. Williams's "treatment group" and "control group" together account for just 6.6 percent of the revenue of the developers

---

218. Williams Report ¶¶75, 77 ("[O]n January 1, 2018 Google began charging a 15% service fee for subscriptions lasting more than one year[.]"). *See also* GADGETS 360, *Google Play Lowers App Subscription Fee to 15 Percent, Matches Apple's Offering*, Oct. 20, 2017, *available at* "https://gadgets360.com/apps/news/google-play-app-subscription-fee-30-percent-to-15-1764923 ("Google said this week it is lowering the amount of money it charges app developers for mobile subscriptions processed through the Google Play app store from 30 percent to 15 percent in a move to better compete with Apple's offering for iOS developers. Developers will be able to make use of the price revision, which goes into effect beginning January 2018, only among their customers who have subscribed to their app for a year. So, for the first year, Google will still take a 30 percent cut on the mobile subscription transactions, the company clarified. This is the same policy Apple employs for auto-renewed paid apps.").

219. Williams Report ¶75; ¶¶84-88; ¶¶111-124.

220. *Id.* ¶86.

221. *Id.*

222. *Id.*

223. Dr. Williams limits his analysis to this time period because, beginning in July 2021, Google introduced a new policy lowering take rates to 15 percent for developers' first $1 million in annual revenue, which he believes would contaminate his "natural experiment." Williams Report ¶116.

224. *Id.* ¶112. Dr. Williams' DID regressions also include fixed effects by month. *Id.*

225. For example, suppose there are only two "products," the first in the "treatment group" and the second in the "control group," and suppose each is initially priced at $5. If price of the first product drops to $4 in January 2018 and the price of the second product remains the same, the DID regression will measure this as pass-through. If the prices of both products fall by the same amount, the DID regression will measure zero pass-through. If the price of the second product rises relative to the first product, the DID regression will also measure this as pass-through.

226. Williams Report ¶117, n. 146.

227. *Id.*

he purports to analyze during his "treatment period."

111. Dr. Williams, like Dr. Burtis, ignores all long-run effects of lower take rates by construction: Although Dr. William's "treatment period" technically runs for 42 months, the vast majority of his data set disappears within the first four to six months of the "treatment period."[228]

B. Neither Dr. Burtis Nor Williams Can Reliably Measure Pass-Through In A More Competitive But-For World With Substantially And Permanently Lower Take Rates for All or Almost All Developers

112. The pass-through analyses presented by Drs. Burtis and Williams cannot capture the long-run effect of permanently lower take rates for all or almost all developers flowing from increased competition. In a competitive world with more than one app store or payment processor, developers would be incentivized to pass on savings from a lower take rate via steering and discounting, to induce consumers to switch to the low-cost provider. In a competitive but-for world, this would be facilitated by (1) multi-homing and steering among competing app stores in the Android App Distribution Market; and (2) consumer (or developer) choice of payment processors with steering in the In-App Aftermarket.[229] This incentive is completely absent in the pass-through analyses of Drs. Burtis and Williams. Developers that enjoyed Google's take-rate decreases did not have to share any of the savings with their customers in order to realize the cost savings.

113. Drs. Burtis and Williams thus mistakenly attempt to study pass-through independently of the competitive mechanism that would engender lower take rates and thus motivate pass-through in the first place. In the competitive but-for world, there would be robust competition in both the Android App Distribution Market and in the In-App Aftermarket. This competition—with steering by developers driving take rate reductions—is completely absent in the pass-through analyses presented by Drs. Burtis and Williams, both of whom rely on narrow and artificial take-rate changes imposed by Google.

114. Rather than measuring pass-through, Drs. Burtis and Williams instead provide compelling evidence of price "stickiness."[230] For example, Dr. Burtis finds that over 90 percent of the products she examined "exhibited no change in price at all during the class period."[231] This does not demonstrate that prices would have remained the same in a more competitive but-for world.

228. *See* Figures 6-7, *infra*.
229. Singer Report ¶¶169-175; ¶¶229-232.
230. Dr. Burtis herself emphasizes price stickiness. Burtis Report ¶¶25-26; ¶151; ¶327.
231. *Id.* ¶175. *See also* Williams Report ¶¶80-81; Figure 3.

What it does demonstrate is that developer pricing exhibits stickiness, which arises due to well-understood behavioral economic phenomena such as consumer anchoring.[232]

115. As explained in the Singer Report, price stickiness would facilitate lower prices in the but-for world.[233] When a new App (or a new form of In-App Content) is developed, a profit-maximizing developer selects a price that maximizes expected profit over the long run, taking into account costs incurred over the long run.[234] To ensure a sufficient rate of return on its investment, a developer faced with the prospect of paying 30 percent of its revenue to Google in perpetuity will (all else equal) need to charge a higher price to consumers than a developer facing a lower take rate. Price stickiness implies that the initial price chosen for an App (or In-App Content) will influence subsequent pricing, and hence reinforces developers' incentives to select an initial price that takes all costs (including the take rate) into account. Because developer costs would have been permanently and substantially lower due to lower take rates, prices would have been permanently and substantially lower for all or almost all developers.[235] Thus, lower take rates would influence developer pricing from the inception of their Apps (or In-App Content). The pass-through analyses of Drs. Burtis and Williams are incapable of measuring this effect.

C. Drs. Burtis and Williams Do Not Follow Standard Econometric or Statistical Methods

116. The pass-through analyses of Drs. Burtis and Dr. Williams do not follow standard econometric or statistical methods. As Dr. Burtis concedes, her pass-through analysis does not control for "economic factors."[236] Indeed, Dr. Burtis does not employ any econometric methods or statistical tests when conducting her pass-through analysis.[237] Dr. Burtis includes no control variables, nor any control group at all her analysis—because the Challenged Conduct is always present in the actual world, she does not (and cannot) observe the prices that would have prevailed in the absence of the Challenged Conduct. This implies that Dr. Burtis's analysis "has little to say about causation."[238] According to standard economic principles, a "before-after" study must

232. *See, e.g.,* Amos Tversky & Daniel Kahneman, *Judgment under Uncertainty: Heuristics and Biases*, 184 SCIENCE 1124, 1128 (1974) ("In many situations, people make estimates by starting from an initial value that is adjusted to yield the final answer…different starting points yield different estimates, which are biased toward the initial values. We call this phenomenon anchoring."). *See also*, Andrea Caceres-Santamaria, *The Anchoring Effect*, Federal Reserve Bank of St. Louis (2021), *available at* https://research.stlouisfed.org/publications/page1-econ/2021/04/01/the-anchoring-effect ("[I]t's the initial price a consumer is exposed to that becomes a consistent reference point when shopping around. The tendency for a person to rely heavily on the first piece of information they receive when making decisions is known as the anchoring effect…Anchoring plays a role in decisions that involve numerical values such as prices…Retailers are very aware that price anchors are an effective tool they can use in their pricing strategy.").

233. Singer Report ¶¶226-227.

234. *Id.* ¶226.

235. Even if one assumes that some developers would not lower their prices in the competitive but-for world, consumers still would benefit from quality improvements in Apps and In-App Content that developers would be able finance due to lower costs from lower take rates. Standard economics shows that competition drives firms to make competitive investments in product quality to keep pace with rivals. *Id.* ¶233 (citing Department of Justice & Federal Trade Commission, *Horizontal Merger Guidelines* (2010), §10).

236. Burtis Report ¶177.

237. For example, Dr. Burtis employs no econometric methods or statistical tests in Table 5, Exhibit 35, Exhibit 50, or Exhibit 55.

238. David Kaye & David Freedman, *Reference Guide on Statistics*, in REFERENCE MANUAL ON SCIENTIFIC EVIDENCE (Federal Judicial Center 2000), at 95.

include a clean "after" period in which the conduct in question is absent.[239] Dr. Burtis lacks a clean "after" period, which invalidates her pass-through analysis.[240]

117. Drs. Burtis and Williams violate the most fundamental assumption of any natural experiment—that the control group is *unaffected* by the policy change at issue.[241] Both ignore the elementary economic fact that a developer does not set prices for one product independently of other products sold by the same developer.[242] For example, Dr. Williams includes Tinder "subscription products" in both his control group and his treatment group. More generally, 75 percent of the expenditures in Dr. Williams's "treatment group" are on Apps that also appear in Dr. Williams's "control group." In the case of Dr. Burtis (who has no control group nor any econometric control variables), 100 percent of the Apps in Dr. Burtis's after period also appear in the before period. Dr. Burtis has no way of controlling for factors, such as price stickiness, that will influence pricing in both time periods. Nor can Dr. Williams control for price stickiness, which will prevent prices in the "treatment group" from diverging from those in the "control group."

118. Relatedly, Drs. Burtis and Williams also ignore that competition among developers makes their pricing interdependent, which further contaminates their pass-through analysis. To illustrate, suppose there are two competing subscription developers and one is in Dr. Williams's "control group" and the other is in his "treatment group." If the developer in the "treatment group" drops its price, the developer in the "control group" will likely do the same. But this will register as zero pass-through in Dr. Williams's DID regressions, which picks up pass-through only when prices in the "treatment group" fall relative to the "control group." Conversely, if the developer in the "control group" does not drop its price (because its take rate is still 30 percent), the developer in the "treatment group" faces a diminished incentive to drop its own price.

D. Drs. Burtis and Williams Ignore the Elementary Economics of Subscription Pricing, Which Incentivizes Introductory Discounts at the Front End

119. Drs. Burtis and Williams both attempt to analyze the Play Store's 2018 take rate reduction for subscription developers, which applied only to customers with a subscription in place

---

239. *See, e.g.,* Justin McCrary & Daniel Rubinfeld, *Measuring Benchmark Damages in Antitrust Litigation* 3(1) JOURNAL OF ECONOMETRIC METHODS 63 (2014) ("the benchmark approach evaluates prices only in the market at issue, comparing prices in the impact period to available prices before and/or after the alleged period of impact (the 'control period')").

240. Dr. Williams also lacks a clean "after" period.

241. *See, e.g.,* JEFFREY WOOLDRIDGE, INTRODUCTORY ECONOMETRICS: A MODERN APPROACH, at 453 (THOMPSON 4TH ED. 2009) ("A natural experiment occurs when some exogenous event—often a change in government policy—changes the environment in which individuals, families, firms, or cities operate. A natural experiment always has a control group, which is not affected by the policy change, and a treatment group, which is thought to be affected by the policy change.")

242. This form of pricing interdependence forms the economic basis for merger analysis. *See* Department of Justice & Federal Trade Commission, *Horizontal Merger Guidelines* (2010), §1. *See also* Steven Berry, *Estimating Discrete Choice Models of Product Differentiation* 25(2) RAND JOURNAL OF ECONOMICS 242–262 (1994); Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects* 70 ANTITRUST LAW JOURNAL 257 (2002); Aviv Nevo, *Mergers with Differentiated Products: the Case of the Ready-to-Eat Cereal Industry*, 31(3) RAND JOURNAL OF ECONOMICS 395-421 (2000).

for twelve continuous months.[243] Both ignore elementary economics, which shows that a subscription developer will tend to find it optimal to charge lower introductory prices, targeting a pool of relatively price-sensitive customers, and higher prices to loyal subscribers that have revealed a greater willingness to pay for the subscription service.[244] This explains why it is common for consumers to receive subscription offers with low *introductory* rates, which are eventually phased out—frequently after one year.[245] It therefore makes little economic sense for Drs. Burtis and Williams to assume that pass-through can be reliably measured based on "back-end" discounts offered to subscription developers' least price-sensitive consumers.

120. The anomalous manner in which take rates were reduced to subscription developers (only for customers maintaining a subscription for twelve continuous months) undermines another fundamental economic incentive for pass-through: The ability to attract new customers with a lower price, thereby offsetting the revenue losses on existing customers. This key incentive is absent in the actual world, but it would exist in a competitive but-for world with lower take rates across the board.

121. Drs. Williams and Burtis also ignore that, if a consumer has subscribed to a product for a year (or more) and paid the same monthly price, that consumer has revealed a strong willingness to pay for the subscription offering. It would make little economic sense for a developer to target price cuts to its least price-sensitive customers. Yet Drs. Burtis and Williams naively assume that pass-through would occur in exactly this way.

E. Drs. Burtis and Williams Ignore That the Play Store's Developer Interface Likely Made It Difficult or Impossible for Most Developers to Violate the Elementary Economics of Subscription Pricing by Forcing Back-End Discounts

122. Drs. Burtis and Williams ignore that the pricing rules in the Play Store's developer interface likely made it difficult or impossible for most developers to drop prices to subscribers after the first year. According to Google's developer pricing rules, subscription Apps may offer discounts to *new* subscribers.[246] But apart from these introductory discounts, I understand that

243. Burtis Report ¶174, n. 193 ("In 2018, Google reduced the service fee rate on subscription IAPs when the consumer's subscription extended beyond a year.").

244. Standard economics shows that, when feasible, firms can earn more profit when the charge different prices to different groups of customers based on their willingness to pay. *See, e.g.,* DENNIS CARLTON & JEFFREY PERLOFF, MODERN INDUSTRIAL ORGANIZATION (Pearson 4th ed. 2005), Chapter 9.

245. *See, e.g.,* Subscribe, THE WALL STREET JOURNAL, *available at* https://store.wsj.com/shop/us/; *see also* https://www.nytimes.com/subscription ("Save 50% for one year. You can cancel anytime.").

246. *See* Play Console Help, *Create a subscription*, *available at* https://support.google.com/googleplay/android-developer/answer/140504?hl=en#zippy=%2Csubscription-prices%2Cintroductory-prices%2Cprice-changes ("5…**Introductory price**: You can offer new subscribers a discounted price for a specific duration. If you do, this must be within the accepted price range and cost less per day than the original price…With introductory pricing, you can

Google's developer interface makes it difficult or impossible for most developers to lower the price to *existing* subscribers after the first twelve months of their subscription without lowering the price to newer subscribers as well.[247] Indeed, developer testimony indicates that developers were unaware of a mechanism to lower the price to existing subscribers after the first year without lowering the price for all subscribers, including new subscribers or existing subscribers who had subscribed for less than a year.[248] In other words, based on the evidence available at this time, it does not appear that Google's developer-pricing interface or published policies provide a mechanism to implement the form of pass-through contemplated by Dr. Williams's "natural experiment" or by Dr. Burtis's "before-after" analysis of subscription developers. This complication would not arise in a competitive but-for world, given that take rates would be substantially and permanently lower for all or almost all subscribers.

F. Drs. Burtis and Williams Ignore the Vast Majority of Revenues for Subscription Developers Subject to Google's 2018 Take Rate Adjustment, and Rely on Artificially Defined "Products"

123. Dr. Williams claims that his DID regression compares the prices of different "subscription products."[249] To limit his "treatment group" to "products" with take rates of exactly 15 percent after 2017,[250] however, Dr. Williams generates an artificially large number of subcategories, and then selects only a handful of these subcategories to use in his "treatment group." He selects his subcategories based on "Product IDs," which do not correspond to product offerings as they are actually presented to customers. Similarly, for purposes of analyzing "Google Play's 2018 service fee reduction for certain subscriptions,"[251] Dr. Burtis limits her analysis of subscription developers to only those "SKUs for which the monthly service fee rate for that SKU fell to 20% or lower and remained at that level for at least three consecutive months."[252] In the process, both ignore the vast majority of revenue for these developers.

---

specify an initial price that applies to a set number of days, weeks, months, or billing periods. For example, you can offer a subscription for $1 per month for the first three months. Or, you can offer an introductory price of $1 for 10 days, followed by a regular monthly price. At the end of the introductory period, users are charged the full subscription price…Introductory prices must be within the accepted price range and less than the subscription's full price. If you're offering a free trial and introductory price, your users are charged the introductory price at the end of the trial. A user can only receive an introductory price to a specific subscription product (SKU) one time. If the introductory period is a different length of time than the subscription period, the introductory price must cost less per day than the original price. For example, if a subscription costs $15 per month (or $0.50 per day), a week-long introductory price must cost less than $3.50. For these calculations, a month is always considered to be 30 days.") (emphasis in original).

247. Google's published policy generally does not allow developers to provide discounts to any subset of customers on a specific subscription product with the exception of introductory discounts. *See* Android Developers, Google Play, Play Billing, Sell Subscriptions, *available at* https://developer.android.com/google/play/billing/subscriptions

248. Deposition of Daniel Scalise (March 11, 2022) at 269:9-21 ("Q. Do you know if you can lower the price of subscriptions on the Google Play Store for some users but not others? A. I don't think you can. Q. Okay. So if you wanted to reward users, say, users who had been subscribed for a year or longer, is it your understanding that you wouldn't be able to give them a lower subscription price but still keep the higher subscription price for new users, right? A. That is my understanding, yes, that that would not be possible.").

249. Williams Report ¶96.

250. *Id.* ¶117, n. 146.

251. Burtis Report ¶174.

252. *Id.* n. 193.

124. To restrict his "natural experiment" to what he terms "clean effects of the treatment," Dr. Williams discards all "products" from his "treatment group" that do not have take rates of exactly 15 percent after 2017, as well as all "products" in the "control group" that do not have take rates of exactly 30 percent.[253] In doing so, Dr. Williams generates artificial "products" that are divorced from economic reality. For example, if Dr. Williams's regression data set is to be believed, Tinder, which has just three major subscription categories (Plus, Gold, and Platinum), offers over 512 distinct "subscription products." To conform to his exacting 30-to-15 requirement, Dr. Williams's regression model uses only 38 of these "products." As shown in Table 1 below, Dr. Williams discards from his "treatment group" more than 99 percent of consumer expenditures on Tinder during his "treatment period."

125. More generally, Table 3 shows that consumer expenditures on the "products" in Dr. Williams's "treatment group" constitute a trivial fraction (just 1.5 percent) of consumer expenditures for the developers (and their Apps) that Dr. Williams purports to analyze during his "treatment period."

TABLE 3: DR. WILLIAMS'S "TREATMENT GROUP" INCLUDES ONLY A SMALL FRACTION OF REVENUE FOR THE DEVELOPERS HE PURPORTS TO ANALYZE DURING HIS "TREATMENT PERIOD"

App	Consumer Spend in "Treatment Group"	Total Consumer Spend in App During "Treatment Period"	Percent In Treatment Group Regression
[redacted]	[redacted]	[redacted]	0.4%
[redacted]	$10	$56,090,839	0.0%
[redacted]	$264,047	$45,439,296	0.6%
[redacted]	$120,212	$43,271,161	0.3%
[redacted]	$123,077	$36,429,107	0.3%
[redacted]	$471,765	$33,125,617	1.4%
[redacted]	$1,098,984	$31,857,584	3.4%
[redacted]	$323,225	$31,531,518	1.0%
[redacted]	$642,417	$24,463,161	2.6%
[redacted]	$74,802	$20,428,078	0.4%
All Others	$13,299,224	$603,350,265	2.2%
TOTAL	**$17,228,862**	**$1,135,645,426**	**1.5%**

126. Dr. Burtis, like Dr. Williams, relies on artificially defined "products" extracted from the Play Store's data production to conduct her pass-through analysis. For example, in Table 5 of the Burtis Report, Dr. Burtis claims that only one percent of "subscription developers" showed evidence of pass-through one month after the take rate was reduced.[254] But the subscription "products" that Dr. Burtis analyzes in Table 5 represent only 3.7 percent of the revenue for the Apps

253. Williams Report ¶117, n. 146.

254. Setting aside all of the other flaws in Dr. Burtis's analysis, it is doubtful whether one would expect to observe pass-through over such a short time horizon. A one-month adjustment period bears no relation to the competitive but-for world.

she purports to analyze.[255] Similarly, in Exhibit 50, Dr. Burtis claims that only two percent of "subscription developers" showed evidence of pass-through within six months after the take rate was reduced. The subscription "products" that Dr. Burtis analyzes in Exhibit 50 represent only 7.3 percent of the revenue for the Apps she purports to analyze.[256] It makes no economic sense for Drs. Williams and Burtis to assume that pass-through can be reliably measured by searching for price cuts to artificially constructed products that make up only a trivial fraction of developer revenue.

127. Because Dr. Williams's "products" are artificially constructed, their quantities sold fluctuate wildly and rapidly disappear over the course of Dr. Williams's "treatment period." Figure 4 below plots the number of monthly [redacted] transactions used in Dr. William's regression, as well as the subset of those transactions classified into Dr. Williams's "treatment group." As seen below, the total number of [redacted] transactions in Dr. Williams's regression (seen in the blue line) declines steeply soon after January 2018. The number of [redacted] "products" classified into Dr. Williams's "treatment group" (seen in the orange line) is tiny, never exceeding [redacted] transactions in a month, and falling to just [redacted] monthly transactions by the end of the "treatment period." It makes no economic sense for Dr. Williams to assume, as he does, that pass-through can be reliably measured by searching for price cuts among this minuscule subset of transactions.

255. See Appendix 2, Table A1.
256. See Appendix 2, Table A2.

FIGURE 4: MONTHLY TRANSACTIONS FOR [redacted] "PRODUCTS" IN DR. WILLIAMS'S REGRESSION

[redacted]

128. Similarly, Figure 5 below displays the aggregate number of monthly transactions for the [redacted] "products" used in Dr. Burtis's pass-through analysis. As before, the volume for these artificially defined "products" fluctuates wildly. These "products" amount to at most [redacted] percent of [redacted] revenue in the Play Store over the (one to six month) timeframe when Dr. Burtis searches for pass-through.[257] It is not reasonable to assume, as Dr. Burtis does, that artificially constructed "products" such as these would drive the pricing decisions of a developer such as [redacted]

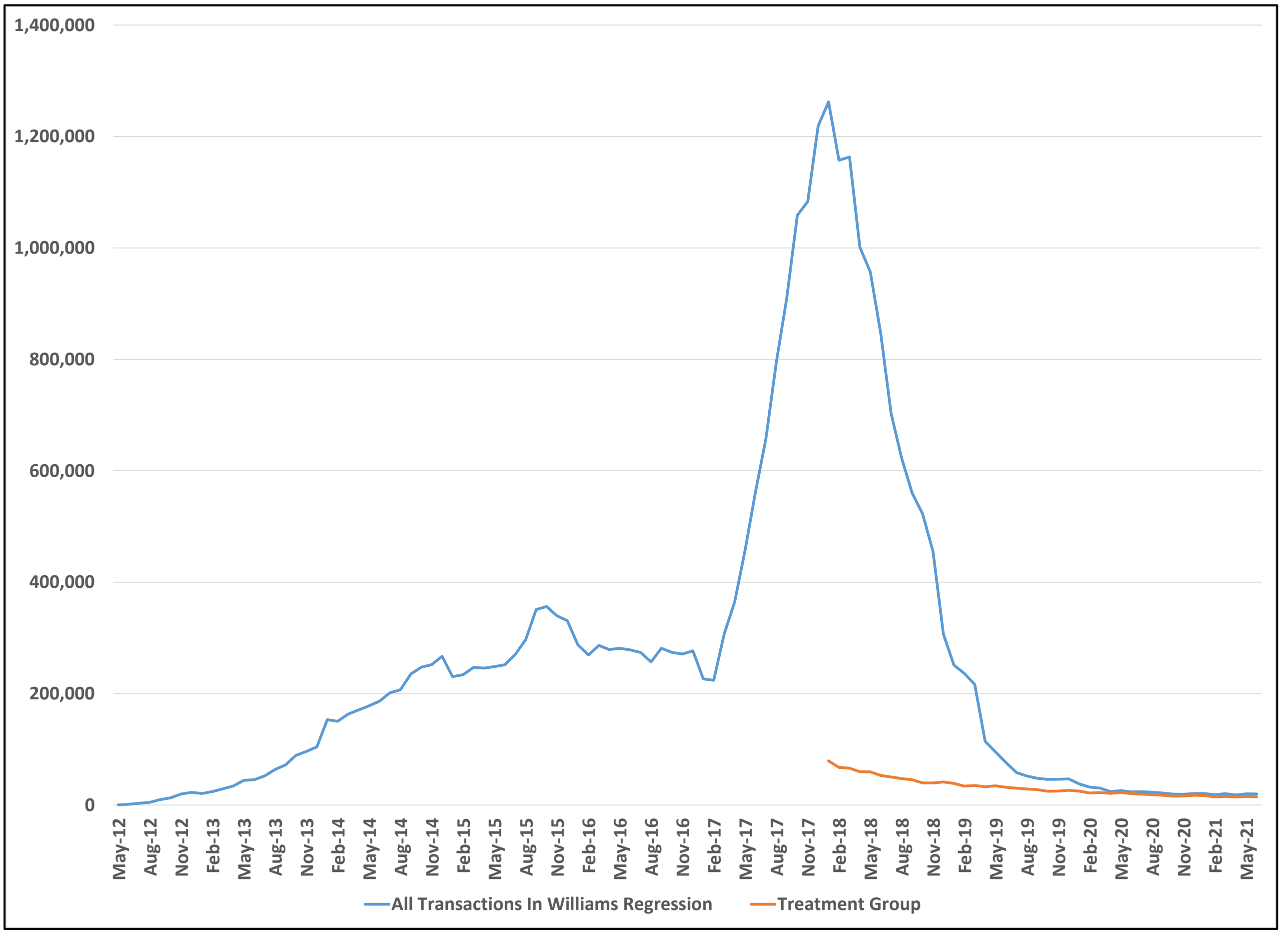
257. See Appendix 2, Tables A1-A2.

FIGURE 5: MONTHLY TRANSACTIONS FOR [redacted] "PRODUCTS" IN DR. BURTIS'S TABLE 5

[redacted]

129. Figure 6 below displays the aggregate number of monthly transactions for all of the "products" used in Dr. Williams's regression. As seen below, the total number of transactions in Dr. Williams's regression (seen in the blue line) declines steeply from its peak in early 2018. The number of "products" classified into Dr. Williams's "treatment group" (seen in the orange line) begins as a small fraction of the blue line, and also falls off steadily. As before, Dr. Williams assumes incorrectly that pass-through can be reliably measured by searching for price cuts among this minuscule subset of transactions. The data underlying his regression model is divorced from economic reality.

FIGURE 6: MONTHLY TRANSACTIONS FOR ALL "PRODUCTS" IN DR. WILLIAMS'S REGRESSION



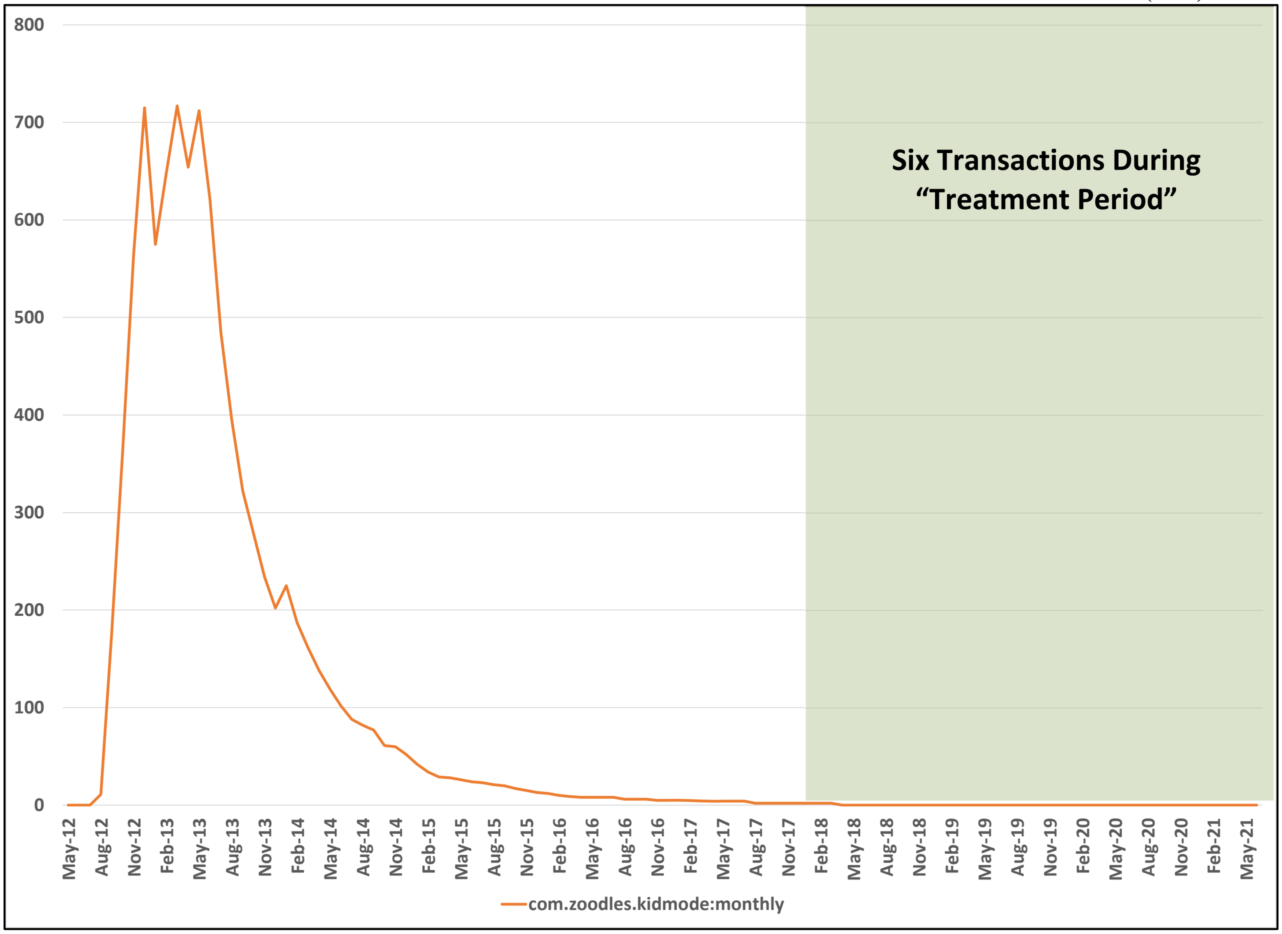
130. The "products" in Dr. Williams's "treatment group" appear only intermittently and with extremely small transaction volumes. During the 42 months of Dr. Williams's "treatment period," from January 2018 through June 2021, the average "product" in the "treatment group" has fewer than five transactions per month. The average "product" also has zero transactions in 31 of the 42 months in Dr. William's "treatment period."[258]

131. The median "product" in Dr. Williams's "treatment group" has just six *total* transactions for *all* 42 months of the "treatment period." Put differently, half of the "products" in Dr. Williams's "treatment group" have six or fewer transactions during the entirety of the "treatment period." Figure 7 below plots the monthly transactions for one such "product." As seen below, this offering had been largely removed from the market before Dr. Williams's "treatment period" had even begun. During the "treatment period," a grand total of six transactions are recorded (two each in January, February, and March of 2018) before the transaction volume drops to zero and stays there.

---

258. Even if one were to ignore all months in which a purported "product" had zero transactions, the average "product" would still have only 20 transactions per month.

FIGURE 7: MONTHLY TRANSACTIONS FOR A "PRODUCT" IN THE "TREATMENT GROUP" WITH THE MEDIAN NUMBER OF TRANSACTIONS DURING THE "TREATMENT PERIOD" (SIX)



132. Similar to Dr. Williams's analysis, the "products" in Dr. Burtis's analysis of subscription developers subject to the Play Store's 2018 take-rate adjustment exhibit instability and appear only intermittently. The median subscription "product" in Dr. Burtis's pass-through analysis had just 22 *total* transactions during the six-month period that she examined. In other words, half of the subscription "products" in Dr. Burtis's pass-through analysis had fewer than 3.7 transactions per month.[259] It makes no economic sense to assume, as Drs. Burtis and Williams do, that artificially constructed products with a *de minimis* number of transactions, and accounting for just a small fraction of developer revenue, would drive developer pricing decisions in response to lower take rates.

133. Because their "subscription products" are a tiny and artificially constructed sliver of the actual products offered by subscription developers, the "subscription products" analyzed by Drs. Burtis and Williams are likely not a representative sample. A representative sample is "a sample judged to fairly represent the population, or a sample drawn by a process likely to give samples that

259. Equal to 22 divided by 6.

fairly represent the population, for example, a large probability sample."[260] As explained above, the subscription "products" analyzed by Drs. Burtis and Williams are defined artificially, constitute a small share of developer revenue, and appear intermittently and in very small (sometimes zero) transaction volumes. The population that the sample is purported to represent should, at a minimum, consist of stable subscription product offerings likely to drive developer pricing decisions.

CONCLUSION

134. For the reasons given above, the opinions that I offered in the Singer Report remain unaltered.

* * *

Hal J. Singer, Ph.D.:

Executed on April 25, 2021.

260. David Kaye & David Freedman, *Reference Guide on Statistics*, REFERENCE MANUAL ON SCIENTIFIC EVIDENCE 295 (3rd ed. National Academies Press 2011).

APPENDIX 1: MATERIALS RELIED UPON

BATES DOCUMENTS

AMZ-GP_00003257

GOOG-PLAY-000005203.R

GOOG-PLAY-000076766

GOOG-PLAY-000076773

GOOG-PLAY-000292207.R

GOOG-PLAY-000294117.R

GOOG-PLAY-000297309.R

GOOG-PLAY-000303918.R

GOOG-PLAY-000336574

GOOG-PLAY-000355570.R

GOOG-PLAY-000416245

GOOG-PLAY-000443763

GOOG-PLAY-000560564

GOOG-PLAY-000568027

GOOG-PLAY-000579868.R

GOOG-PLAY-001501104

GOOG-PLAY-001507837

GOOG-PLAY-001556407

GOOG-PLAY-002414772

GOOG-PLAY-003605103

GOOG-PLAY-006990552

GOOG-PLAY-007203251

GOOG-PLAY-007346993

GOOG-PLAY-007745829

GOOG-PLAY-007879368.C

GOOG-PLAY-007887261

GOOG-PLAY-008162331

GOOG-PLAY4-007109523

GP MDL-TMO-0002416

GP MDL-TMO-0131438

DEPOSITIONS

Deposition of Jamie Rosenberg (Feb. 10, 2022)

Deposition of Daniel Scalise (March 11, 2022)

LITERATURE

Alexander Galetovic, Stephen Haber, Lew Zaretzki, *An estimate of the average cumulative royalty yield in the world mobile phone industry: Theory, measurement and results*, 42(3) Telecommunications Policy (2018)

Amos Tversky & Daniel Kahneman, *Judgment under Uncertainty: Heuristics and Biases*, 184 Science (1974)

Anindya Ghose & Sang Pil Han, *Estimating Demand for Mobile Applications in the New Economy*, 60(6) Management Science (2014)

Aviv Nevo, *Mergers with Differentiated Products: the Case of the Ready-to-Eat Cereal Industry*, 31(3) RAND Journal of Economics 395-421 (2000)

Christopher Knittel & Konstantinos Metaxoglou, *Estimation Of Random-Coefficient Demand Models: Two Empiricists' Perspective* 96(1) Review of Economics and Statistics 34 (2014)

David Evans, Two-Sided Market Definition in Market Definition In Antitrust: Theory And Case Studies (ABA Section of Antitrust Law) 1-35 (2009), *available at* papers.ssrn.com/sol3/papers.cfm?abstract_id=1396751

David Kaye & David Freedman*, Reference Guide on Statistics*, Reference Manual on Scientific Evidence (3rd ed. National Academies Press 2011)

Dennis Carlton & Jeffrey Perloff, Modern Industrial Organization (Pearson 4th ed. 2005)

Department of Justice & Federal Trade Commission, *Horizontal Merger Guidelines* (2010), §1

Gregory Werden & Luke Froeb, *The Antitrust Logit Model For Predicting Unilateral Competitive Effects,* 70 Antitrust Law Journal 257 (2002)

Gregory Werden & Luke Froeb, *The Effects of Mergers in Differentiated Products Industries: Logit Demand and Merger Policy* 10(2) Journal of Law, Economics, & Organization (1994)

Herbert Hovenkamp, *Digital Cluster Markets,* Columbia Business Law Review (forthcoming 2022), *available at* https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3820062

Jean-Charles Rochet & Jean Tirole, *Platform Competition in Two-Sided Markets,* 1(4) Journal of the European Economic Association (2003)

Jeffrey Wooldridge, Introductory Econometrics: A Modern Approach (Thompson 4th ed. 2009)

Jerry Hausman & Greg Leonard*, Efficiencies from the Consumer Viewpoint,* 17(3) George Mason Law Review (1999)

José Azar, Ioana Marinescu, & Marshall Steinbaum, *Labor Market Concentration*, 57(3) JOURNAL OF HUMAN RESOURCES (2020)

Judith Holdershaw, Philip Gendall and Ron Garland, *The Widespread Use of Odd Pricing in the Retail Sector*, Marketing Bulletin (8), (1997), *available at* http://marketing-bulletin.massey.ac.nz/V8/MB_V8_N1_Holdershaw.pdf

Justin McCrary & Daniel Rubinfeld*, Measuring Benchmark Damages in Antitrust Litigation* 3(1) Journal of Econometric Methods (2014)

N. Gregory Mankiw, Principles of Microeconomics (Cengage Learning 8th ed. 2018)

Nathan Miller, Conor Ryan, Marc Remer, & Gloria Sheu*, Approximating the Price Effects of Mergers: Numerical Evidence and an Empirical Application* DOJ Economic Analysis Group Discussion Paper 12-8 (2012)

Nathan Miller, Marc Remer, & Gloria Sheu, *Using cost pass-through to calibrate demand*, 118 Economics Letters (2013)

Steven Berry, *Estimating Discrete Choice Models of Product Differentiation* 25(2) RAND Journal of Economics 242–262 (1994)

William Landes & Richard Posner, *Market Power in Antitrust Cases,* 94(5) Harvard Law Review (1981)

PUBLICLY AVAILABLE MATERIALS

"(Im)Balance of Power: How Market Concentration Affects Worker Compensation and Consumer Prices," Testimony to the House Committee on Economic Disparity and Fairness in Growth (Apr. 6, 2022)

Android Developers, Google Play, Play Billing, Sell Subscriptions, *available at* https://developer.android.com/google/play/billing/subscriptions

AT Internet Institute, *Are we heading towards the end of Internet Explorer's reign in Europe?* (July 27, 2010), *archived at* https://web.archive.org/web/20100806153329/http://www.atinternet-institute.com/en-us/browsers-barometer/browser-barometer-june-2010/index-1-2-3-205.html

Chance Miller, *Paddle unveils 'first alternative' to Apple's App Store In-App Purchase system following Epic ruling*, 9to5Mac (Oct. 7, 2021), *available at* https://9to5mac.com/2021/10/07/app-store-iap-paddle-system-announcement/

David Curry, *App Data Report,* Business of Apps (2022)

Discover, Discover it Cash Back Card, *available at* https://www.discover.com/credit-cards/cash-back/it-card.html

Epic Games Store, FAQ, *available at* https://www.epicgames.com/site/en-US/epic-games-store-faq

Gadgets 360, *Google Play Lowers App Subscription Fee to 15 Percent, Matches Apple's Offering* (Oct. 20, 2017), *available at* "https://gadgets360.com/apps/news/google-play-app-subscription-fee-30-percent-to-15-1764923

Gartner Press Release, *Gartner Says Worldwide PC Shipments Declined 5% in Fourth Quarter of 2021 but Grew Nearly 10% for the Year* (Jan. 12, 2022), *available at* https://www.gartner.com/en/newsroom/press-releases/2022-01-12-gartner-says-worldwide-pc-shipments-declined-5-percent-in-fourth-quarter-of-2021-but-grew-nearly-10-percent-for-the-year

Georgina Tzanetos, *Netflix Loses 31% Market Share as Streaming Rivals Gain Loyal Subscribers*, Yahoo! (April 7, 2021), *available at* https://www.yahoo.com/video/netflix-loses-31-market-share-204537722.html

Google Play Store, *Choose a category and tags for your app or game*, *available at* https://support.google.com/googleplay/android-developer/answer/9859673?hl=en

Google Play Store, Apps, *available at* https://play.google.com/store/apps

The New York Times, *Unlimited access to all the journalism we offer, available at* https://www.nytimes.com/subscription

In-App Purchase, paddle, *available at* https://www.paddle.com/platform/in-app-purchase

James Brumley, *Netflix is Losing Market Share, but This is the Actual Risk to Shareholders*, The Motley Fool (Apr. 25, 2021), *available at* https://www.fool.com/investing/2021/04/15/netflix-is-losing-market-share-but-thats-not-the-a/

James W. Cortada, *How the IBM PC Won, Then Lost, the Personal Computer Market - Not even Big Blue could keep up with its creation's success*, IEEE Spectrum (July 21, 2021), *available at* https://spectrum.ieee.org/how-the-ibm-pc-won-then-lost-the-personal-computer-market

Joe Wituschek, *Apple TV+ gains market share in the United States while Netflix loses it*, JustWatch (Jan. 24, 2022), *available at* https://www.imore.com/apple-tv-gains-market-share-united-states-while-netflix-loses-it

Nielsen, The Total Audience Report Q4 2014, *available at* https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/total-audience-report-q4-2014.pdf

PayPal Fee Changes (Aug. 2, 2021), *available at* https://www.paypalobjects.com/marketing/ua/pdf/US/en/feepages-080221.pdf

Paypal Help Center, *How can I update my payment preferences for micropayments?, available at* https://www.paypal.com/gf/smarthelp/article/how-can-i-update-my-payment-preferences-for-micropayments-faq1691

Play Console Help, Choose a category and tags for your app or game, *available at* https://support.google.com/googleplay/android-developer/answer/9859673?hl=en#zippy=%2Capps%2Cgames

Play Console Help, Create a subscription, *available at* https://support.google.com/googleplay/android-developer/answer/140504?hl=en#zippy=%2Csubscription-prices%2Cintroductory-prices%2Cprice-changes

Play Console Help, Supported Locations for Distribution to Google Play Users, accessed Dec. 27, 2021, *available at* https://web.archive.org/web/20211227224037/https:/support.google.com/googleplay/android-developer/answer/10532353?visit_id=637762416354084080-1400722469&rd=1

Play Console Help, Supported Locations for Distribution to Google Play Users, accessed Feb. 18, 2022, *available at* https://web.archive.org/web/20220218131358/https:/support.google.com/googleplay/android-developer/answer/10532353?visit_id=637807868385671271-2942202130&rd=1

PriceIntelligently, Odd-Even Pricing, *available at* https://www.priceintelligently.com/odd-even-pricing#

Sameer Samat (Google Vice President, Product Management), *Boosting developer success on Google Play* (Mar. 16, 2021), *available at* https://android-developers.googleblog.com/2021/03/boosting-dev-success.html

Sameer Samat (Google Vice President, Product Management), *Evolving our business model to address developer needs* (Oct. 21, 2021), *available at* https://android-developers.googleblog.com/2021/10/evolving-business-model.html

Secure an Android Device, *available at* https://source.android.com/security/overview

Sensor Tower, 2021 – 2025 Mobile Market Forecast (2021), *available at* go.sensortower.com/rs/351-RWH-315/images/Sensor-Tower-2021-2025-Market-Forecast.pdf

shopify, Odd-Even Pricing, *available at* https://www.shopify.com/encyclopedia/odd-even-pricing

Statcounter GlobalStats, *Browser Market Share Worldwide* (October 2021), *available at* https://gs.statcounter.com/browser-market-share#monthly-202110-202110-bar

Statista, Market share of global general purpose card brands American Express, Diners/Discover, JCB, Mastercard, UnionPay and Visa from 2014 to 2020, based on number of transactions, *available at* https://www.statista.com/statistics/278970/share-of-purchase-transactions-on-global-credit-cards/

Statista, Most popular Google Play app categories as of 1st quarter 2021, by share of available apps, *available at*: https://www.statista.com/statistics/279286/google-play-android-app-categories/

Subscribe, The Wall Street Journal, *available at* https://store.wsj.com/shop/us/

Testimony for the U.S. Senate Judiciary Committee, Subcommittee on Competition Policy, Antitrust, and Consumer Rights, Jared Sine, Chief Legal Officer, Match Group (April 21, 2021)

Thecounter.com, Browser Stats (April 2004), archived at https://web.archive.org/web/20111101195133/http://www.thecounter.com/stats/2004/April/browser.php

Top Paid Apps, Google Play, *available at* https://play.google.com/store/apps/collection/cluster?clp=0g4jCiEKG3RvcHNlbGxpbmdfcGFpZF9BUFBMSUNBVElPThAHGAM%3D:S:ANO1ljLdnoU&gsr=CibSDiMKIQobdG9wc2VsbGluZ19wYWlkX0FQUExJQ0FUSU9OEAcYAw%3D%3D:S:ANO1ljIKVpg

TRIAL MATERIALS

Complaint, In re Google Play Consumer Antitrust Litigation, No. 3:20-CV-05761-JD (Dec. 20, 2021)

Expert Report Of Douglas Craig Schmidt, PhD (February 28, 2022)

Expert Report of Dr. Michelle M. Burtis (March 31, 2022)

Expert Report of Hal J. Singer, Ph.D. (February 28, 2022)

Expert Report of Michael Williams (February 28, 2022)

APPENDIX 2: ADDITIONAL TABLES AND FIGURES

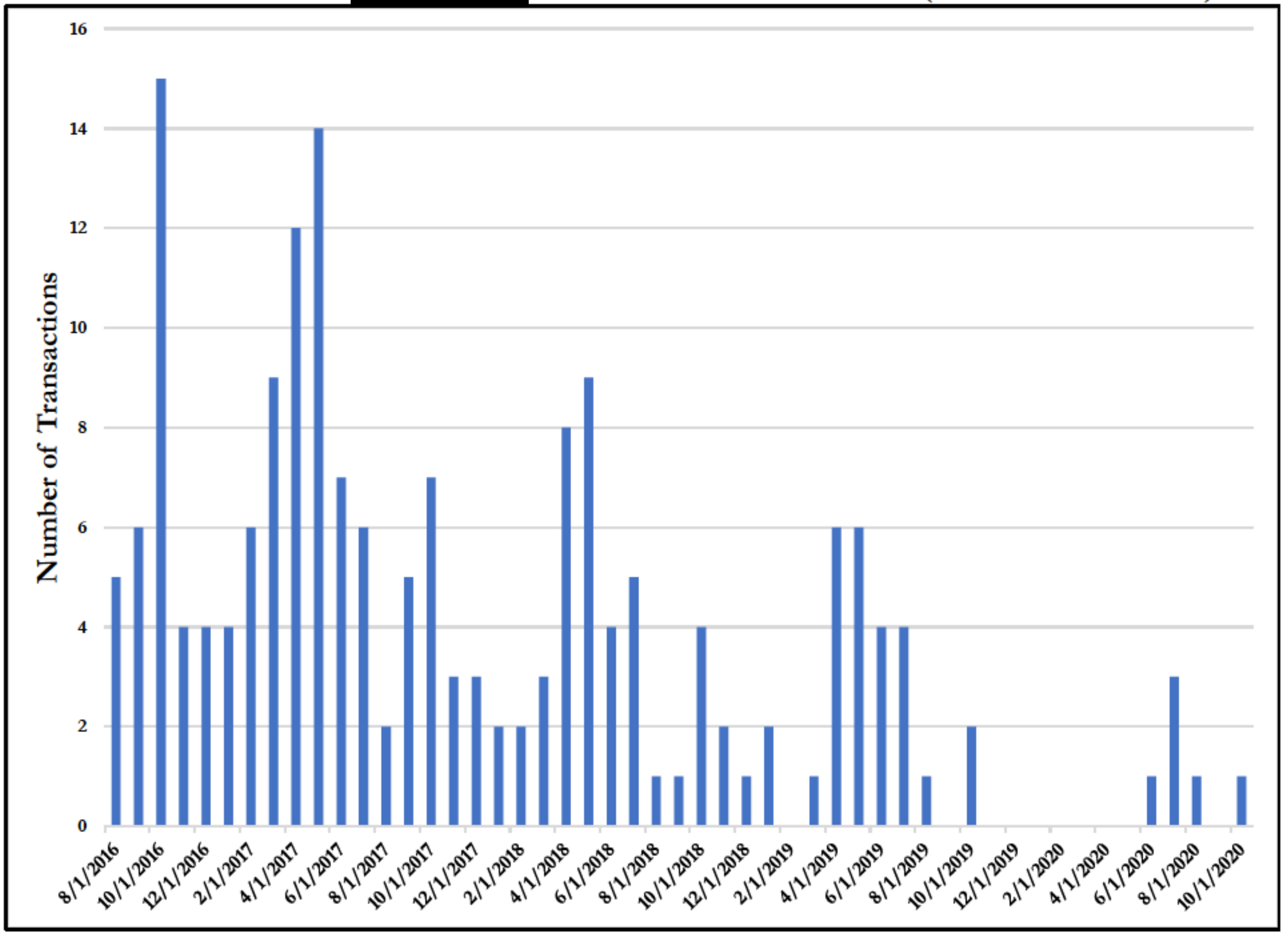
APPENDIX TABLE A1: THE "SUBSCRIPTION DEVELOPERS" IN DR. BURTIS'S TABLE 5 REPRESENT ONLY A SMALL PROPORTION OF REVENUE FOR THE DEVELOPERS SHE PURPORTS TO ANALYZE

App	Consumer Spend in Burtis's Table 5	App Total Consumer Spend One Month After Service Fee Rate Drop	Percent of Consumer Spend in Burtis's Table 5
[redacted]	$ [redacted]	$ [redacted]	0.6%
	$ 96,426	$ 20,280,402	0.5%
	$ 363,406	$ 17,505,717	2.1%
	$ 150,468	$ 12,676,827	1.2%
	$ 693,580	$ 12,520,837	5.5%
	$ 761,630	$ 10,112,602	7.5%
	$ 500,277	$ 8,987,715	5.6%
	$ 41,001	$ 7,319,714	0.6%
	$ 21,987	$ 6,544,597	0.3%
	$ 43,035	$ 6,264,122	0.7%
All Others	$ 14,723,817	$ 242,246,005	6.1%
TOTAL	**$ 18,332,152**	**$ 492,769,479**	**3.7%**

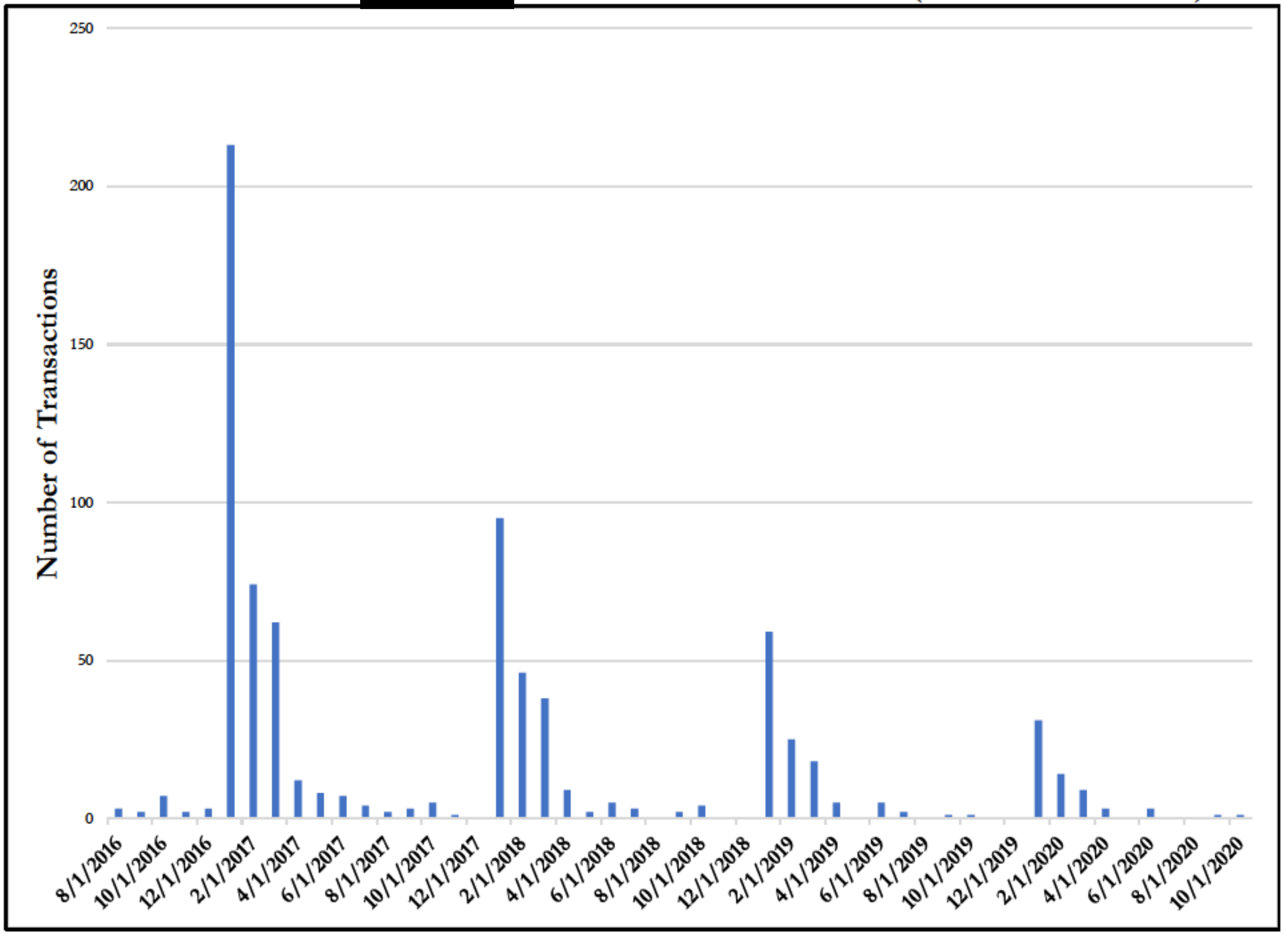
APPENDIX TABLE A2: THE "SUBSCRIPTION DEVELOPERS" IN DR. BURTIS'S EXHIBIT 50 REPRESENT ONLY A SMALL PROPORTION OF REVENUE FOR THE DEVELOPERS SHE PURPORTS TO ANALYZE

App	Consumer Spend in Burtis's Exhibit 50	App Total Consumer Spend Six Months After Service Fee Rate Drop	Percent of Consumer Spend in Burtis's Exhibit 50
[redacted]	$ [redacted]	$ [redacted]	2.4%
	$ 3,373,216	$ 37,938,150	8.9%
	$ 621,392	$ 35,772,224	1.7%
	$ 2,058,768	$ 33,988,047	6.1%
	$ 3,584,467	$ 32,547,692	11.0%
	$ 449,804	$ 31,869,465	1.4%
	$ 2,918,197	$ 25,365,315	11.5%
	$ 1,661,622	$ 20,644,346	8.0%
	$ 2,636,188	$ 19,001,394	13.9%
	$ 529,139	$ 18,273,621	2.9%
All Others	$ 66,840,769	$ 752,064,205	8.9%
TOTAL	**$ 89,883,137**	**$ 1,225,516,498**	**7.3%**

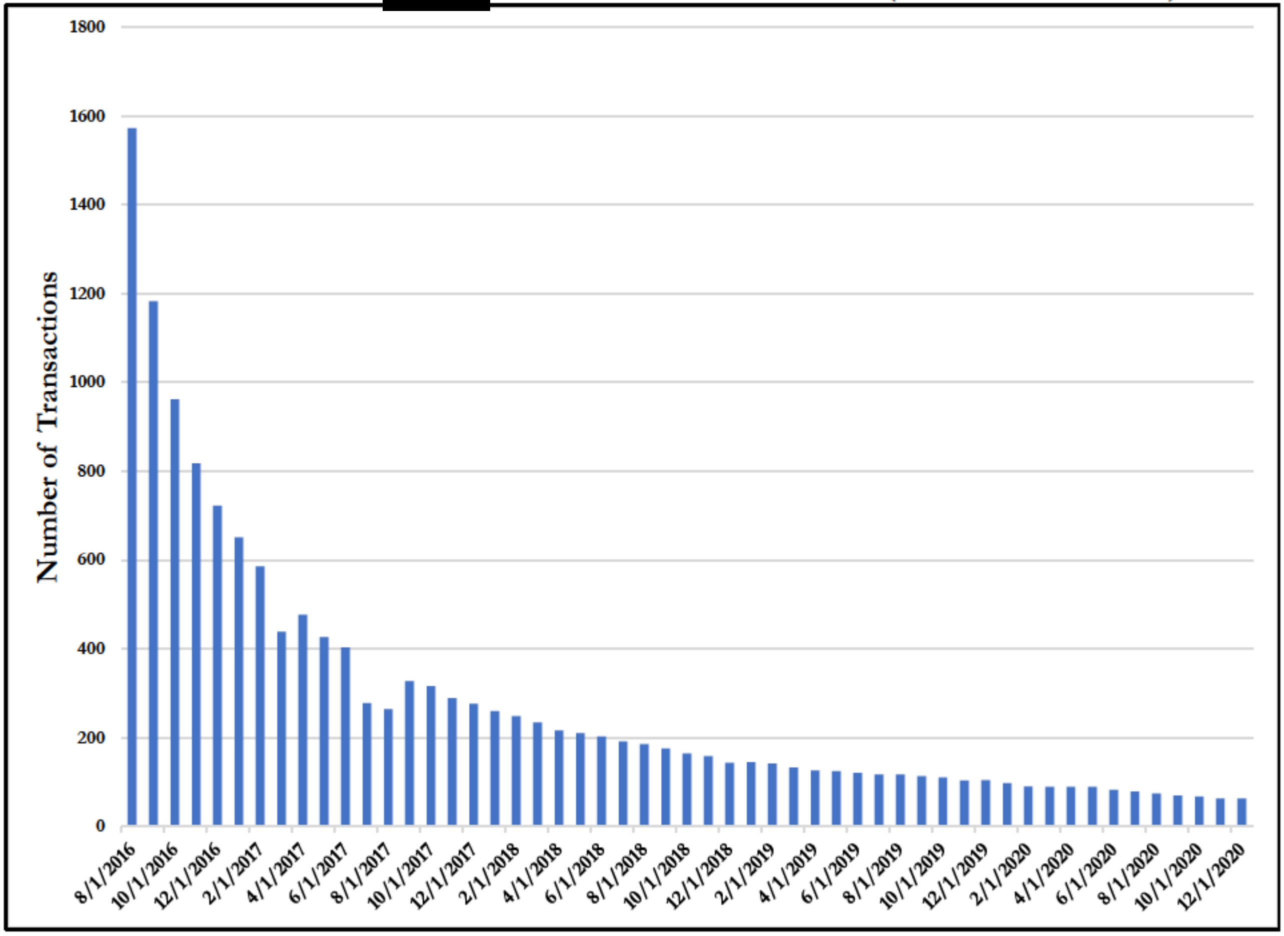
APPENDIX FIGURE A1: [redacted] SUBSCRIPTION TRANSACTIONS (8/1/2016 – 10/1/2020)



APPENDIX FIGURE A2: [redacted] SUBSCRIPTION TRANSACTIONS (8/1/2016 – 10/1/2020)



APPENDIX FIGURE A3: [redacted] SUBSCRIPTION TRANSACTIONS (8/1/2016 – 12/1/2020)



REDACTED VERSION

Exhibit A4
to
C. Cramer Declaration

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

IN RE GOOGLE PLAY STORE ANTITRUST LITIGATION THIS DOCUMENT RELATES TO: *In re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD	No. 3:21-md-02981-JD **EXPERT REPORT OF DOUGLAS CRAIG SCHMIDT, PH.D.** Judge: Hon. James Donato

HIGHLY CONFIDENTIAL UNDER PROTECTIVE ORDER

TABLE OF CONTENTS

I. INTRODUCTION

Android powers more mobile devices globally than any other mobile software system. The earliest versions of Android were open source, drawing on the talents of global developers to build and augment the platform. After acquiring Android, however, Google has moved functionalities from open-source Android to Google's proprietary version of Android ("Google Android"). And it has implemented a number of technological impediments that artificially limit the ability of developers to distribute, and users to obtain, applications (referred to herein as "apps") outside of the Google Play Store. These technological impediments also artificially limit the ability of users to download and use app stores that potentially compete with the Google Play Store. Counsel for Mary Carr, Daniel Egerter, Zack Palmer, Serina Moglia, Matthew Atkinson, and Alex Iwamoto, on behalf of themselves and all others similarly situated (the "Consumer Plaintiffs") and counsel for Pure Sweat Basketball Inc.; Peekya App Services, Inc.; LittleHoots, LLC; and Scalisco LLC d/b/a Rescue Pets, on behalf of themselves and all others similarly situated (the "Developer Plaintiffs"), have asked me to lay out the basic components of the Android mobile software system and its associated tools, the means through which software apps are accessed and installed on mobile devices, and the mechanics through which billing and payment processing systems allow users to purchase both apps and digital or physical content and products from within the app.

In this Report, I draw upon my knowledge and experience in software engineering and system design along with the information cited herein to address the questions: (1) whether Google has imposed technological restraints that artificially restrict or inhibit consumer access to competitive app stores and to apps sold through competitive app stores or other distribution channels, and (2) whether there are technical reasons for Google's billing service product,

Google Play Billing, to be included in the flow of purchases made directly in apps that users have downloaded to their mobile device.

I am submitting this report at the class certification stage of this case, and I understand that discovery is ongoing. I reserve the right to supplement and to add new information and opinions to this report. I also reserve the right to provide additional information and opinions in the merits report I will be submitting at a later stage of this case.

II. QUALIFICATIONS

I am the Cornelius Vanderbilt Professor of Engineering in the Department of Electrical Engineering and Computer Science at Vanderbilt University. I am an experienced software engineer, with a focus on technology development in the areas of middleware platforms and model-driven tools for high-performance and real-time apps. I have worked in industry, for the government, and in academia. As a Visiting Scientist at Carnegie Mellon University's Software Engineering Institute, I have worked with the Ultra-Large-Scale Systems team to define the challenging problems, promising technology areas, and research roadmaps for the national R&D effort on shaping future innovations in complex software-reliant systems. As the Deputy Director of the Defense Advanced Research Projects Agency, an agency within the Department of Defense, I helped to set and guide the National IT research and development agenda. In my role as the Associate Provost of Research Development and Technologies at Vanderbilt, I am charged with developing cohesive and sustainable information technology services to advance research and scholarship across Vanderbilt's ten schools and colleges.

I am an author or co-author of over 400 peer-reviewed publications, reports, books, and book chapters. I currently conduct research on patterns, optimizations, and experimental analysis of advanced software techniques that facilitate the development of distributed real-time and embedded middleware and model driven architectures running over high-speed networks and

interconnects. I have been deposed extensively and have testified as an expert in the federal courts, principally in the context of patent litigation involving software and middleware systems. A copy of my curriculum vitae, including a list of my testimony over the last four years, is attached to this report as Exhibit A.

I have no stake in the outcome of this case. I am being compensated for my work in this case at the rate of $550 per hour. The materials I relied upon in forming my opinions are summarized in Exhibit B. I reserve the right to supplement my opinions as more information becomes available.

III. GENERAL TECHNOLOGICAL BACKGROUND

A. High-Level Technical Background Regarding Mobile Devices, Operating Systems, and Apps

To understand how mobile apps are installed and function on devices running Android, it is helpful to have a general understanding of mobile devices, their software, and how apps are built.

Mobile devices, such as smart phones and tablets, are essentially hand-held portable computers. These devices can connect to the internet over cellular and/or wi-fi networks. Like other computing devices, they are controlled by an operating system ("OS"), which is software that manages the device's hardware and software resources and, in some cases, provides interfaces through which third-party app developers can create programs that run on the mobile device.

While the OS provides the basic software to manage a mobile device, enabling functionality like network connectivity, battery and power management, and other core features, most functionality is provided by apps, which are software programs designed to perform a variety of tasks. Once pre-loaded or installed on the device, apps that are designed to be accessed

by the user of the device ("user-facing" apps) are typically displayed on the device screen through a representative image, known as an icon. The user opens or "calls" a user-facing app through tapping or clicking on the icon. Unless I indicate otherwise, when I refer to an "app" in this report, I am referring to a user-facing app.

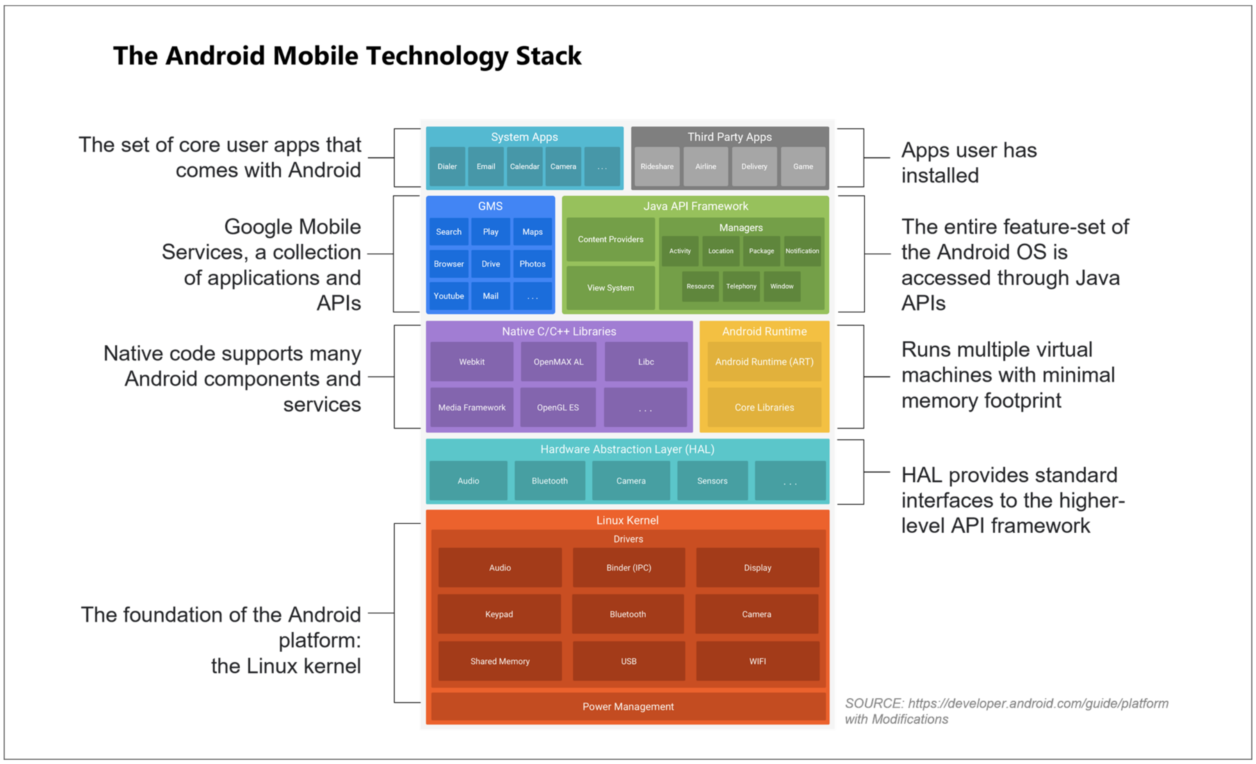
Modern software systems can be thought of as a series of layers, with each layer building on the layer below it. Developers of each layer leverage the functionality and features of the lower layers to save time and develop software more reliably. Each layer exposes an app programmer interface ("API") that defines the set of features and functions provided by the layer. Layers that sit between the core operating system components and user-facing apps are often called "middleware." Middleware is software that provides common services and capabilities to apps beyond the functionalities offered by the OS. Essentially functioning as a hidden translation layer, middleware enables communication and data management for apps and is not meant to be accessible to the average retail user.

Developers of user-facing apps can incorporate into their apps external software packages that allow their apps to interact with, and borrow from, (a) other apps and middleware on the device that perform specific functionalities, and (b) services that may be provided by external servers through internet connections. This integration allows the developer to utilize pre-existing software rather than having to build each component from scratch. These software packages also expose APIs that define what functions are available and how those functions can be leveraged to achieve the desired outcome.

Another important component of mobile device systems is a Software Development Kit ("SDK") which is a set of tools that developers can use when building apps. An SDK allows

developers to build software that uses a pre-existing API or set of APIs. Some examples of SDKs include the Android SDK, the Google Play Services SDK, and the Stripe Payments SDK.

In sum, the software on mobile devices is typically composed of an OS, middleware, and apps. Developers create apps using SDKs. Apps utilize APIs exposed by the OS, middleware, and other apps. The following graphic visually demonstrates these concepts.



B. Users Face Substantial Costs When Switching Between Google Android and iOS

Google Android and Apple iOS are the two most prominent mobile software systems, and each environment leverages different programming languages, programming paradigms, frameworks, and capabilities. The Android and iOS mobile software systems are not compatible, so software developers must create independent versions of their apps to operate on each system. After a user has elected to purchase a device on a particular mobile platform, such as Google Android, they are effectively locked-in to the platform for that device since users face substantial

costs to switch between Google Android and other mobile device platforms. As Google has noted, “Switching phones = switching ecosystems. You not only have to learn how to use a new OS, but completely new platforms around it.” GOOG-PLAY-000880576.R at 589.R.

Users face several switching costs relating to apps. For example, Google Android pushes users towards Google’s proprietary apps, including the Google Play Store. App downloads and purchases made within the Google Play Store might not be transferrable to Apple iOS devices, in which case a user must find them and download or purchase them again from Apple’s App Store. A user may also need to transfer the data in their apps and recalibrate their settings to achieve a comparable experience on the new platform. It is possible to lose such app-related data and purchases.

If the user acquires a new phone within the Google Android ecosystem, their apps and app purchases remain available and, in some cases, may automatically download. This automated capability, however, is not available with a switch from Google Android to an Apple iOS device. Switching from one platform to another is therefore not as simple as buying a different phone but instead involves reconstructing the user’s digital life. Users also face costs associated with transferring non-app data, such as photos and videos, from a mobile device on one platform to a mobile device on another platform. In most cases this requires multiple steps that a user must manually undertake and some formats of data may not be supported on the new device on the new platform. There is also the possibility of loss of such data. Indeed, Google noted that losing data is one of the biggest concerns that users have with switching to a new platform, along with apps/feature incompatibility and having to learn a new OS. GOOG-PLAY-000880576.R at 580.R.

In addition, even a successful transfer of data and apps does not equal a successful switch since a user must still adapt to the new phone. GOOG-PLAY-000880576.R at 589.R. For example, the user interfaces for an Apple iOS device and a Google Android device function differently and leverage different design paradigms (e.g., iOS devices have lacked a physical home button since the iPhone 8 and have never had a "back" button; whereas, many Google Android devices still possess the back and home buttons as soft keys on the touchscreen of the device). These differences require significant expenditures of time and effort by users to learn and master. Google has concluded that this "feels a lot like learning a foreign language." *Id.*

Switching can also be a complicated and time-consuming process. At one point, switching from an Apple iOS device to a Google Android device involved an average of 40 steps and could take as long as 9 hours. GOOG-PLAY-000880576.R at 589.R. While both Google and Apple have made attempts to streamline this process, it is still something that a user must figure out how to do and then manually implement.

Finally, there are also costs arising from the fact that devices on one platform are typically compatible with each other, but not fully compatible with devices on other platforms. For example, if a user purchases an app using his or her Apple iPhone, that app can usually also be accessed on his or her other Apple devices, like an Apple iPad and or Apple Watch, and messages a user sends from his or her iPhone using Apple's iMessage app will show up across those devices as well as his or her Apple computers. But if a user decides to switch to a Google Android phone, the Google Android phone will not necessarily interoperate and synchronize with his or her Apple devices and certain capabilities, such as placing or receiving calls from his or her Apple Watch, may be lost altogether. There are also hardware implications with respect to accessories and other ancillary products. For example, Google's PixelBuds (wireless

headphones), while broadly compatible with any Bluetooth audio device, provide the greatest functionality when attached to a Google Android device. Someone leaving the Google Android platform and going to the Apple iOS platform would lose access to the greatest functionality (*i.e.,* real-time language translation).

C. A Brief Explanation of Android Software

Android originally was developed as "open source" software, meaning that anyone could inspect, modify, or enhance the source code to add to or manipulate the Android software functionality. The Android Open-Source Project ("AOSP") was started in 2003. It was created to guide development of Android software for the mobile platform. The Android software consists of OS, middleware that enables additional functionality for apps, and a set of integrated apps. Google purchased Android, including rights to AOSP, in 2005.

After purchasing AOSP, Google developed and implemented a set of requirements for what it deems "Android Compatible Devices." Android Compatible Devices are defined by Google's Android Compatibility Program. The Android Compatibility Program consists of three components: the AOSP source code (OS, middleware, and integrated apps); the Compatibility Definition Document ("CDD") promulgated by Google; and passing Google's Compatibility Test Suite ("CTS").[1]

Android Compatible Devices fall into two main categories. The first category of devices runs only AOSP source code as defined and maintained by Google—it is entirely free, does not require the user to enter into a licensing agreement, and may be used by anyone. In this report, I refer to the AOSP source code simply as "AOSP."

---

[1] See Android – Developers, Device compatibility overview, available at https://developer.android.com/guide/practices/compatibility; Android – Source, Android Compatibility Program Overview, available at https://source.android.com/compatibility/overview.html.

The second category of devices runs the AOSP-based software system I refer to as "Google Android," which is licensed by Google to manufacturers of mobile devices (referred to herein as "original equipment manufacturers" or "OEMs") and includes the AOSP source code and a package of software referred to as Google Mobile Services ("GMS") that provides additional middleware and user-facing apps.[2] I discuss GMS in more detail below. When I discuss the features and functionality of AOSP, those features and functionalities are also included in Google Android. However, the reverse is not necessarily true—the features and functionality of Google Android are not necessarily included within AOSP.[3]

A mobile device can also run AOSP-based software but not comply with Google's Android Compatible Device requirements. In such instances, an OEM or other developer may have modified the AOSP source code to suit its commercial purposes. Such AOSP-based software does not satisfy the requirements of Google's CDD and will not pass Google's CTS. I refer to such AOSP-based software as "Forked Android Software" or "Forked Android."

In the United States, the most well-known Forked Android devices are the Amazon Fire product line. In China, Forked Android systems are the most popular mobile software systems because the Chinese government does not allow mobile devices sold there to have Google's Mobile Services pre-installed.

D. GMS is a Key Difference Between AOSP and Google Android

As I mention above, the second category of "Android Compatible Devices" runs the mobile software I refer to as "Google Android." While Google Android is based on AOSP,

[2] Google from time to time also makes modifications to the AOSP source code and adds other software to create various versions of Google Android.

[3] A third category of Android Compatible Devices uses software based upon AOSP but does not include GMS. Since these devices are primarily used in specific industrial and commercial apps rather than consumer mobile devices, I do not address them in this report.

Google additionally includes GMS in Google Android. GMS is composed of middleware and a suite of user-facing apps. Google's inclusion of GMS in Google Android results in significant differences between AOSP and Google Android.

1. GMS includes middleware that is unique to Google Android

GMS adds middleware components to Google Android that do not exist in AOSP. These components are composed of APIs plus supporting independent sets of executable code. Google refers to this middleware and the services it provides as "Google Play Services."[4] As middleware, Google Play Services connects user facing apps, the services provided within Google Play Services, and the AOSP OS. Google Play Services is present as an app on all Google Android devices. [5]

Google Play Services runs in the background of a Google Android device at all times and manages a wide range of functionalities. Developers can build apps that access the services in Google Play Services through the Play Services SDK. The Play Services SDK provides a different set of APIs than the standard open-source set available within AOSP.

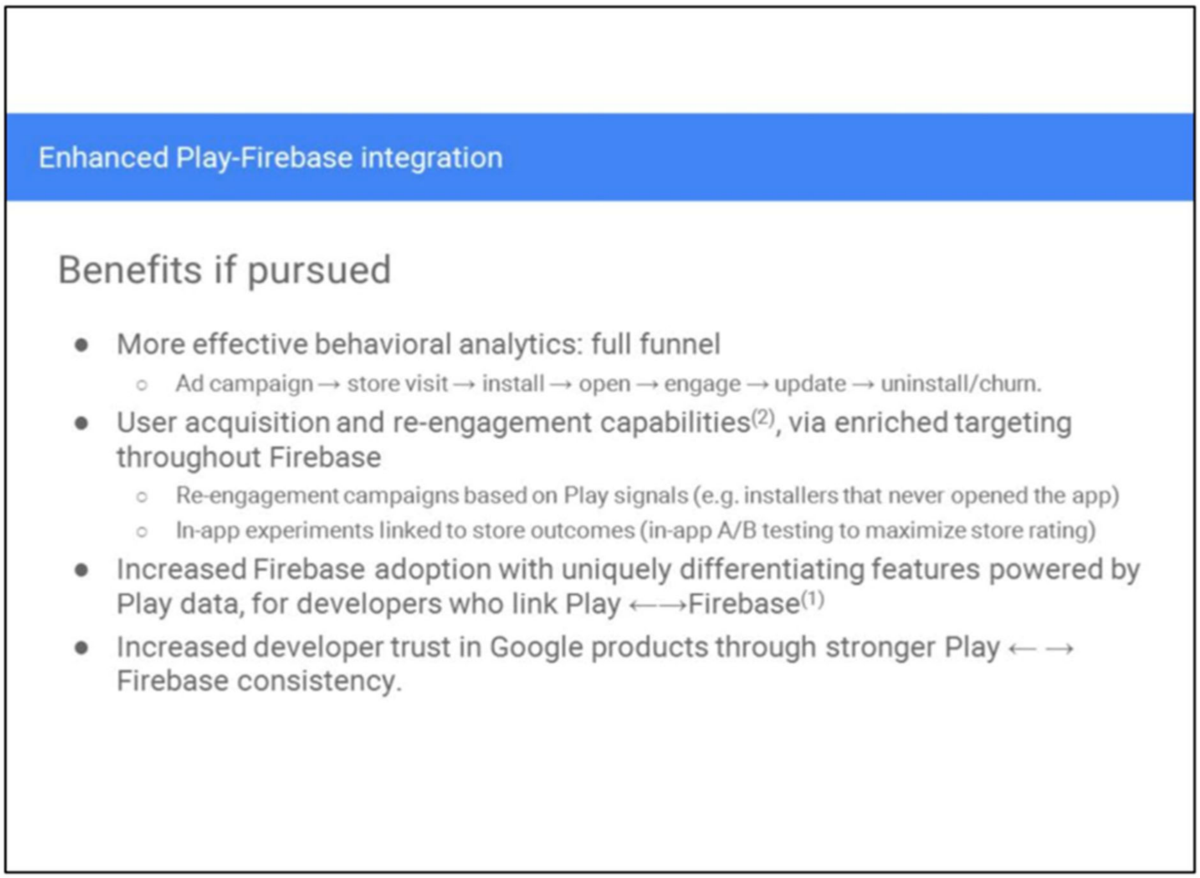
For example, Google Play Services includes an enhanced location services framework called the Fused Location Provider[6] that can provide user-facing apps with better and more precise location data than the original services within the AOSP mobile software system. Google Play Services also includes other APIs and services, including those for mobile advertising (*e.g.,* AdMob); analytics (*e.g.*, Google Analytics for Firebase); and identity (*e.g.*, sign-in with Google).

[4] Google Play Services is not to be confused with the Google Play Store. Technologically the Google Play Store need not be on a device in order for Google Play Services to provide its functionality.

[5] As I mention in Footnote 3, there are Android Compatible Devices whose software does not include GMS. Since these devices primarily have special commercial or industrial uses and are not typically consumer mobile devices, I do not address them in this report.

[6] https://developers.google.com/location-context/fused-location-provider

Google collects a substantial amount of user data through Google Play Services. It shares this user data with developers of apps that rely on Google Play Services. Google Play Services also provides an invaluable data stream to Google about how users interact with apps on their platform and gives Google unique insights into events and user behaviors that are not visible to third-parties:[7]



Specifically, through the use of Google Play Services, Google is able to track [redacted] a valuable metric that provides insight: "[t]hird party SDKs cannot [redacted]

[7] GOOG-PLAY-000343365.R

[redacted]" GOOG-PLAY-000343365.R at 396.R. Moreover, Google is able to drive additional advertising revenue by targeting users based on their interactions with apps installed on their device: "[redacted] Enrich targeting throughout Firebase (e.g. FCM [Firebase Cloud Messaging], config)." (*Id.* at 396.R – 397.R)

2. GMS includes a suite of apps that are not included in AOSP Android

While the Google Play Services part of GMS provides additional middleware components that are less obvious to users, GMS also provides a suite of Google-proprietary apps that are not present on AOSP. Most importantly for the purposes of this report, this GMS suite of apps includes the Google Play Store. But it also includes Google Search, Gmail, Maps, Chrome, YouTube and multiple additional apps. The GMS suite of apps, in turn, exposes APIs that developers can use to incorporate the functionality of the GMS apps within the developers' apps.

This GMS suite of apps, including the Google Play Store, is present on every Google Android mobile device.[8] In most instances, the apps in the GMS suite replace the open-source alternative apps included in AOSP. Google does not publish the source code for the apps in the GMS suite. This restriction is a departure from the early versions of Google Android. In those versions, the bulk of the Google Android app functionality was provided by the open-source AOSP apps with Google providing only modest or optional Google-specific enhancements, such as an enhanced connection to Gmail, with their proprietary apps. Over time, however, Google

[8] As I mention in Footnote 3, there are Android Compatible Devices which use software that does not include GMS. Since these devices primarily have special commercial or industrial uses and are not typically consumer mobile devices, I do not address them in this report.

has favored development of their GMS suite of apps over the development of the open-source apps in AOSP.

For example, Google identified that using location services to get "the best location results" was "very tedious." (Google I/O 2013 – What's New in Google Play Services = https://www.youtube.com/watch?v=49pWckcaZEI at 3:34). Rather than improving location services in AOSP, so that those improvements would be in both AOSP and Google Android, Google improved the location services only in Google Android. Google replaced the location services in Google Android with its Fused Location Provider that makes determining a user's location "nice and easy." *Id*. However, Google does not make this improved location provider available to AOSP devices, meaning that apps relying upon it are tightly coupled to Google Android. Apps using the Fused Location Provider will therefore crash if not properly designed to run on AOSP, which would take a significant amount of additional development work.

3. The inclusion of GMS in Google Android means that apps developed to run on Google Android will not function properly on AOSP

Apps developed to run on Google Android using APIs exposed by GMS will not function properly on devices running on AOSP or Forked Android because GMS (and hence the APIs it exposes) is not present on those devices.

For example, if a developer wishes to use the functionality of Google Maps from the GMS app suite, that developer builds an app that includes calls to the Google Maps APIs exposed by GMS. These API calls allow users to access a Google map within the developers' app without separately opening Google Maps. Examples of apps that access the Maps APIs in the GMS app suite include Uber, McDonalds, and Starbucks. These apps explicitly call to the Google Maps APIs exposed in GMS, so if GMS is not present on the device the app is running on, the app will crash. Therefore, apps that utilize these GMS components, such as Google Maps,

are tightly coupled to Google Android devices and are not usable on devices running AOSP or Forked Android Software.

Similarly, apps developed using the Play Services SDK that call to APIs that access services provided by the Google Play Services middleware will not function properly on devices running on AOSP or Forked Android because the Google Play Services App (and hence the Google Play Services APIs) is not present on those devices. For example, using Google's enhanced location services requires explicit calls to Google Play Services APIs that are not present in AOSP. The services accessed by calling these enhanced location APIs leverage nearby WiFi networks, cell towers, and other radio signals to more accurately determine a user's location compared to using GPS alone. This feature is only available on Google Android and provides significantly more accurate location information more rapidly than using the built-in GPS receiver, and allows apps like Uber to reliably locate the user to provide services. Apps that leverage this and other Google Play Services APIs will not function properly on devices that run software without those APIs, including devices that run AOSP and Forked Android Software. To implement an alternative, developers must first identify an alternative source of the equivalent functionality and write code to access that functionality.

With its inclusion of GMS, Google has effectively created an entirely new platform that is incompatible with AOSP. When there are multiple APIs and frameworks in use, the cost of migrating from one set of frameworks to another can quickly increase to the point that it rivals the cost of developing an entirely new app. Apps that are written for Google Android with GMS are not compatible with AOSP Android. They are not the same platform, and an app that leverages APIs from Google Android cannot target AOSP Android and non-Google Forked Android at the same time.

4. Google could provide developers with SDKs for AOSP allowing them to use the functionalities of GMS without the suite being on the device, as Google has done for iOS apps

As with devices running AOSP Android and Forked Android, devices running Apple's iOS do not have GMS installed on the device. However, Google provides developers with an SDK that allows them to build code into their iOS apps that eliminates the need for GMS. Using these Google-provided SDKs, developers can build iOS apps that call a given functionality over the internet that would otherwise be provided by GMS on a Google Android device.

For example, the Starbucks app on a Google Android device is built to use Google Maps functions by calling to APIs exposed by the Google Maps app on the device. Therefore, Google Maps and GMS must be on the device for the Starbucks app to function on any Android device. However, a Starbucks app on an iOS device is built using an SDK provided by Google to call the Google Maps functionality over the Internet, so neither Google Maps nor GMS need be on the iOS device for the Starbucks app to call the Google Maps functionality. As a consequence, developers seeking to create apps for AOSP or Forked Android devices face significant hurdles that they simply do not encounter when writing for Google Android or iOS—developers must both identify an alternative source of the equivalent functionality and write code to access that functionality.

5. Developers face substantial costs to develop apps for multiple mobile software systems

Developers face substantial costs to develop apps for multiple mobile software systems. For instance, Google Android and Apple iOS are the two most prominent mobile software systems, and each environment leverages different programming languages, programming paradigms, frameworks, and capabilities. While cross platform development solutions exist, they typically involve developing to the lowest common denominator. In the most straight forward

example, iOS and Google Android have completely different guidelines with respect to user-interface expectations. As a result, there is frequently a significant amount of platform-specific design work to ensure that apps feel consistent and "native" to a given platform.[9, 10] To maximize user experience, developers must duplicate efforts across each platform they wish to target since there is no mechanism to automatically make one app built for one system compatible with the other.

Even something as simple as user interface guidelines vary between the Google Android and Apple iOS platforms. A specific example is that Google Android has always had a "back" soft key or button (initially in physical form and now as a soft key on the touch screen of a device) whereas Apple iOS does not. Beyond user interface differences, iOS apps use a completely different set of APIs, services, and frameworks than Google Android. While, for example, the Starbucks app on Google Android may *feel* similar to users as the Starbucks app on iOS, this similarity is not even skin deep. Differences like these require developers to not only write code targeting the language and frameworks of the platform, but also to often customize the designs so they are consistent with platform standards.

In addition to the programming language and design guidelines, the platform APIs and frameworks are an integral consideration when developing an app for a given platform. In fact, understanding the app frameworks, along with the key considerations, assumptions, and capabilities they embody, is essential to make an app functional, and is often more important than understanding programming language features.

[9] https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/

[10] https://material.io/develop/android

For example, the Google Android platform and the Oracle Java ME platform both allow developers to write apps in Java for a mobile environment, but they are completely incompatible with each other. An app written for Google Android will not function in the Java ME environment and vice versa. Therefore, the programming language alone does not dictate compatibility. Moreover, apps that are developed to utilize APIs exposed by GMS on Google Android, such as APIs that provide access to Google Maps or to the enhanced location services of Google Play Services, do not work on platforms that do not contain the GMS. To implement an alternative, developers must first identify an alternative source of the equivalent functionality and write code to access that functionality.

In sum, developers must spend time and resources writing different versions of the same app if they want the app to be usable on devices running different mobile software systems, such as Google Android and Apple iOS.

E. Permissions are a Key Feature of AOSP and Google Android

Apps that run on AOSP or Google Android must have permission to perform an operation. Generally, permissions are divided into two categories: permissions that restrict actions, such as querying a mobile phone's location, and permissions that restrict data, such as accessing a mobile phone's contact database.

In earlier versions of AOSP and Google Android, permission usage was declared by the app developer by including a reference to the permission in the app package. The permission was granted on an all-or-nothing basis: as a condition of installing the app, the user was required to accept all permissions (*e.g.*, the app always had permission to access the location of the phone). Starting in August, 2017 with the release of version 8 of AOSP and Google Android, the systems have given users more nuanced control over the permissions they grant. Apps now request

permissions after installation when they are needed (*e.g.*, only when the user tries to view their location on a map).

Even within this newer system, however, users do not have complete control. Certain permissions cannot be granted or abrogated by the user. Most importantly, the INSTALL_PACKAGES permission can only be granted by OEMs, and only to apps that are preinstalled on the mobile device. Put another way, a retail user cannot grant the INSTALL_PACKAGES permission to an app.

The INSTALL_PACKAGES permission is important because it allows an app, such as an app store, to install other apps without user intervention. So, for instance, an app store must have the INSTALL_PACKAGES permission to install an app on a user's phone automatically without the user going through a manual installation process. In addition, an app store must have the INSTALL_PACKAGES permission to automatically update the apps it installs (as is sometimes necessary for security purposes) without user action.[11]

The INSTALL_PACKAGES permission can only be granted to apps that are preinstalled on the mobile device. If an OEM allowed third-party apps installed by the user to have INSTALL_PACKAGES permissions, the device would not be compatible with Google's CDD and would fail Google's CTS.[12] Moreover, users cannot grant the INSTALL_PACKAGES permission to any apps they install, which means that only the Google Play Store and app stores

[11] Such background updating without user action is referred to as "automatic updating."

[12] See "Android 12 Compatibility Definition," available at https://source.android.com/compatibility/12/android-12-cdd ("[C-0-6] MUST NOT install app packages from unknown sources, unless the app that requests the installation meets all the following requirements: It MUST declare the REQUEST_INSTALL_PACKAGES permission or have the android:targetSdkVersion set at 24 or lower. It MUST have been granted permission by the user to install apps from unknown sources.").

pre-installed by OEMs can install and, and until very recently,[13] update apps automatically. Third-party app stores that are sideloaded outside of the Google Play Store or an OEM app store will necessarily come from a source that does not have INSTALL_PACKAGES permission. Likewise, third-party app stores sideloaded by the user will not have INSTALL_PACKAGES permission.

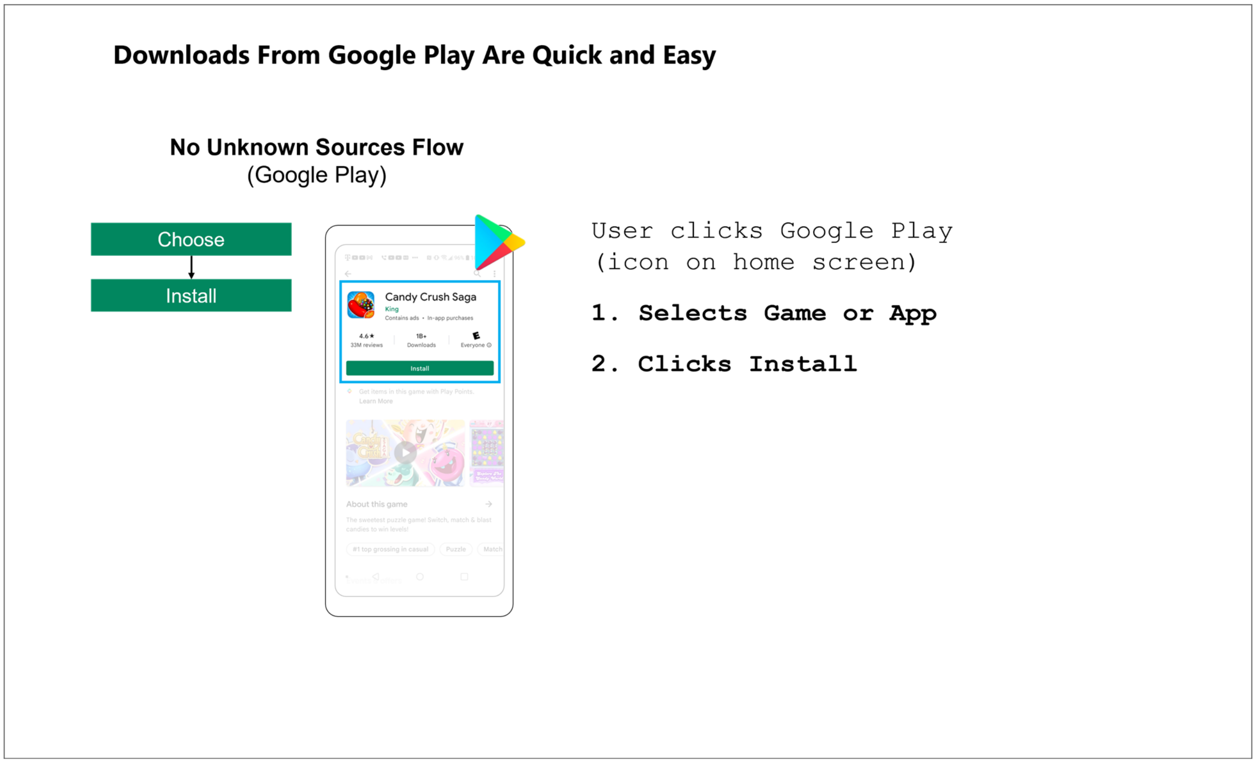
F. Overview of App Installations

From a user perspective, apps can be installed in one of three ways: (1) via the app stores that possess the INSTALL_PACKAGES permission (*e.g.,* the Google Play Store); (2) via the Unknown Sources Flow (for third-party app stores, developer website downloads, peer-to-peer sharing software, or in some cases, backup software); and (3) via the developer tools command line interface.

1. Installation via app stores that possess the INSTALL_PACKAGES permission

The most familiar and user-accessible mechanism for installing apps is via an app store that is installed by the device manufacturer (known as a natively-installed app store) that possesses the INSTALL_PACKAGES permission. These app stores include Google Play Store and other OEM pre-loaded app stores. Native app stores with the INSTALL_PACKAGES permissions can install apps without triggering the unknown sources flow and perform app updates without user intervention.

[13] I note that in October 2021, with the launch of Google Android version 12, Google began allowing users to opt-in to automatic updates for a given store. *See* https://www.xda-developers.com/android-12-alternative-app-stores-update-apps-background/. The user still must opt-in for each app store they wish to allow to install updates automatically and the app store installing the update must be the store that installed the app originally.



As shown in the screen shot above, installing apps via a pre-installed app store, such as the Google Play Store, is a straightforward, simple, and easy process that can be accomplished through a few button taps, all located on the same screen. Moreover, all prompts presented to the user are initiated and controlled by the app store, giving the app store the ability to control the messaging and language displayed to the user. This ability is in contrast to third-party app stores that have no control over the warnings that are displayed during the Unknown Sources Flow or over the prompt that asks the user if they are certain they want to install the app (also called a "speed bump"), that is displayed prior to installation.

2. Installation via the Unknown Sources Flow

Installation via the Unknown Sources Flow stands in stark contrast to installation via app stores that possess the INSTALL_PACKAGES permission. Since the initial version of Android, the Unknown Sources Flow has been the primary option available to users to install those third-party app stores and apps that are not pre-installed or available in the Google Play Store, and

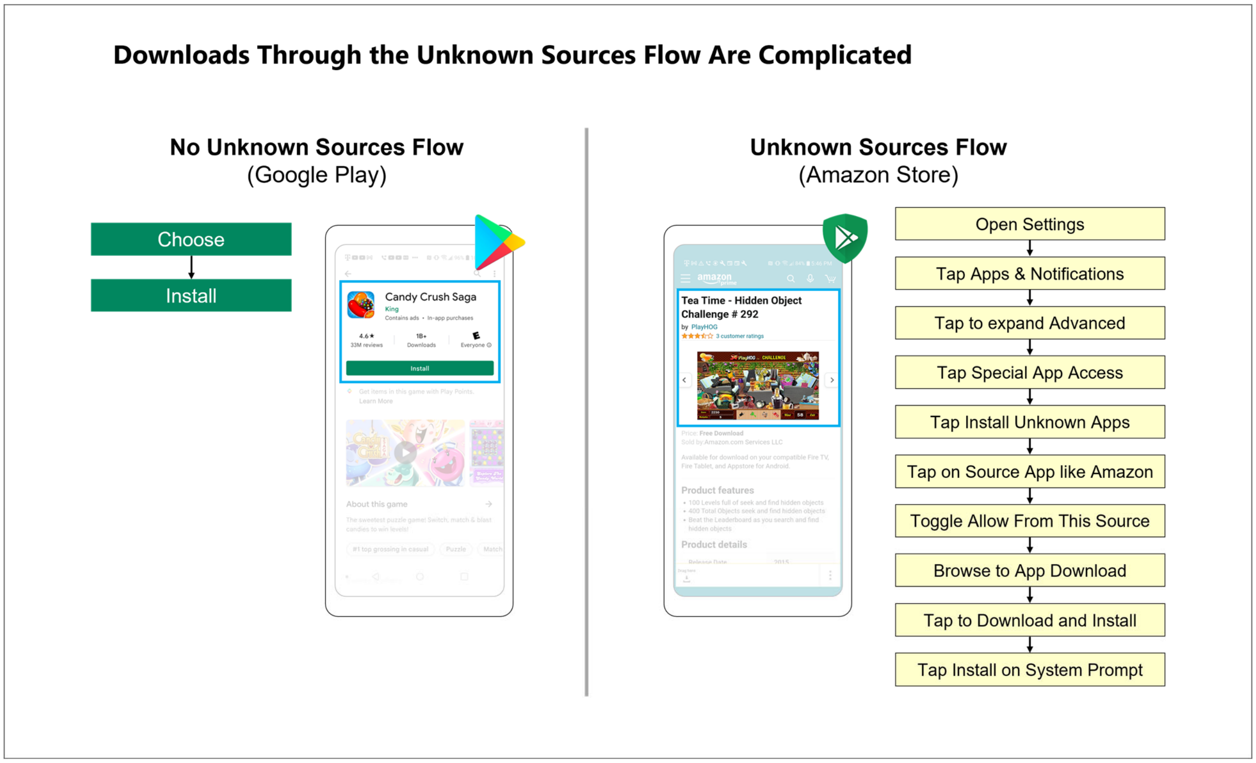
hence come from sources that do not possess the INSTALL_PACKAGES permission. The default settings in AOSP, which are determined by Google, block all installations that are not initiated by sources holding the INSTALL_PACKAGES permission.

The Unknown Sources Flow is time consuming and difficult relative to an install via the Google Play Store. It requires the user to find the third-party app store, download the third-party app store, navigate to a separate settings screen, identify and change the permission toggle in the device settings so that the app store from the “unknown source” can be installed, and install the app store. To install an app from that store, the user must download the app, navigate to yet another separate settings screen, identify and change the permission toggle in the device settings again so that the app from the “unknown source” can be installed, return to the app store, and finally tap to install. This flow is similar for apps downloaded from a developer’s website, without the initial step of downloading the app store itself.

Moreover, at several sub-steps in the sideloading process, the user is confronted with warnings that the app is from an “unknown source” and may be harmful to the user’s phone. Internally, Google uses the softer term “alternate sources” interchangeably with “unknown source,” yet Google only presents these as “unknown sources” to users. See Deposition of David Kleidermacher, February 4, 2022, at 436:16-21: “these terms, like alternate sources and unknown sources, they precede my time. I don't know the original reason why they were created. I have heard them used interchangeably for a long, long time.”

Thus, when the user wishes to install any subsequent apps (including updates for previously installed apps) from the same third-party app store, Google Android, not the app store, presents the user with a “speed bump” prompt further increasing friction. This additional confirmation prompt appears even if the origin of the app is readily identifiable as a reputable

source. Thus, even for downloads from a reputable company, such as Amazon, the Unknown Sources Flow is significantly more complex than downloads from Google Play. For example, the following graphic compares an installation from the Google Play Store (no Unknown Sources Flow) with an installation from the Amazon Store (Triggers the Unknown Sources Flow) in Android 11.



Google refers to this sort of app installation as "sideloading." *See, e.g.,* GOOG-PLAY-004494298.R at 317.R. Moreover, Google refers to the difficulty introduced by the Unknown Sources Flow as "friction." *See, e.g.,* GOOG-PLAY-002011285.R at 288.R. In particular, Google defines "friction (unknown sources)" as occurring when an app store that is a potential competitor of the Google Play Store is not preinstalled. *Id.* Sideloading happens any time a user tries to install an app outside of the Google Play Store or another OEM-installed app store.[14]

[14] In some cases, Google has internally used the term "sideloading" in reference to all off-Play app installs, including apps downloaded from OEM-installed app stores. GOOG-PLAY-000042623.R.

Users can obtain sideloaded apps from developer websites, or from app stores that are not pre-installed on the device. In some countries, such as India, sideloading also occurs when users directly share apps with each other via Bluetooth, which is likely common in part because of poor Internet and bandwidth constraints. *See* GOOG-PLAY-001285448 at 450 (noting that "[i]n emerging markets, where connections are expensive or unreliable, more than 1B users share apps via Peer to Peer apps like SHAREit, Xender, and FilesGo.").

3. Installation via developer tools and the Android Debug Bridge

Finally, users can also install apps using developer tools and the Android Debug Bridge (ADB). This mechanism is designed for developers to install apps during the development process and requires a combination of settings on the phone and utilities on a desktop or laptop computer connected to the phone via a USB cable. While this installation mechanism does work, it is not a viable option for general users to install apps as it requires enabling Developer Settings (a hidden menu accessed by tapping 5 times—or more, depending on the device—on the Android build number in the Settings app), configuring USB debugging, downloading the Android developer utilities, and executing a command line app to install the APK file.

G. Overview of In-App Transactions

Companies have offered products and services for sale via the internet in a secure manner for decades. There are numerous companies and services that exist to facilitate the secure capture of payment information to purchase both physical and digital goods. Generally, an in-app purchase ("IAP") has the following steps:

1. Authorization of payment where the user presents payment information, and the merchant works with the payment processor to determine whether the payment information can cover the full amount of the purchase.

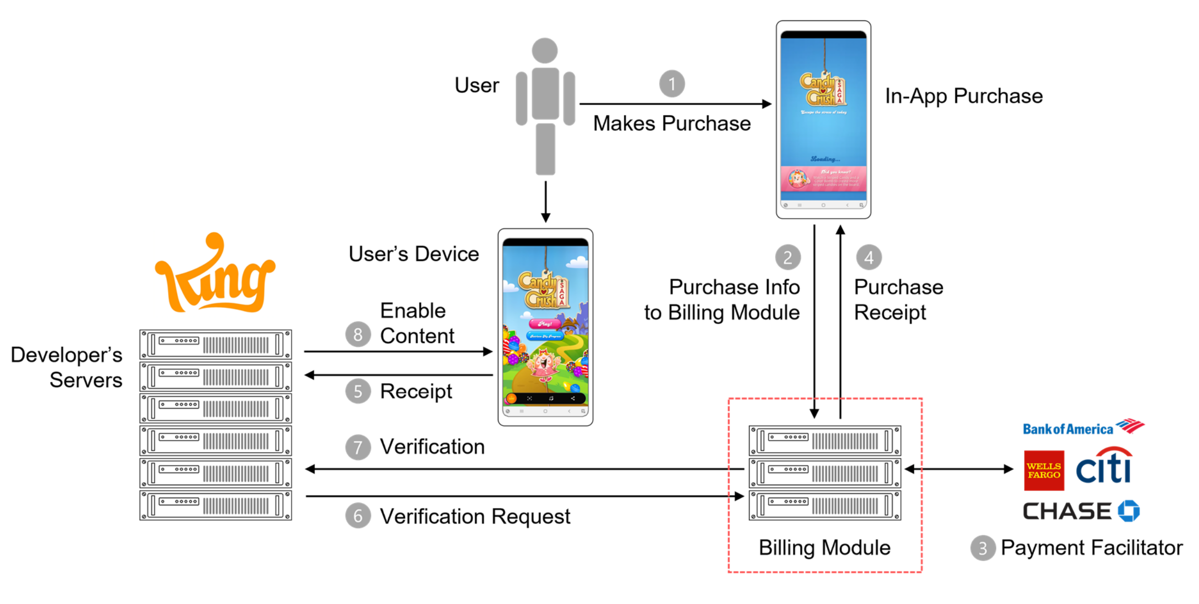
2. Capture of payment information where the merchant instructs the payment processor to debit the user account.

3. Capture of confirmation where the payment processor notifies the merchant the capture was successful. It's at this point that the merchant is typically responsible for updating their internal systems that certain inventory should be allocated to a certain customer.

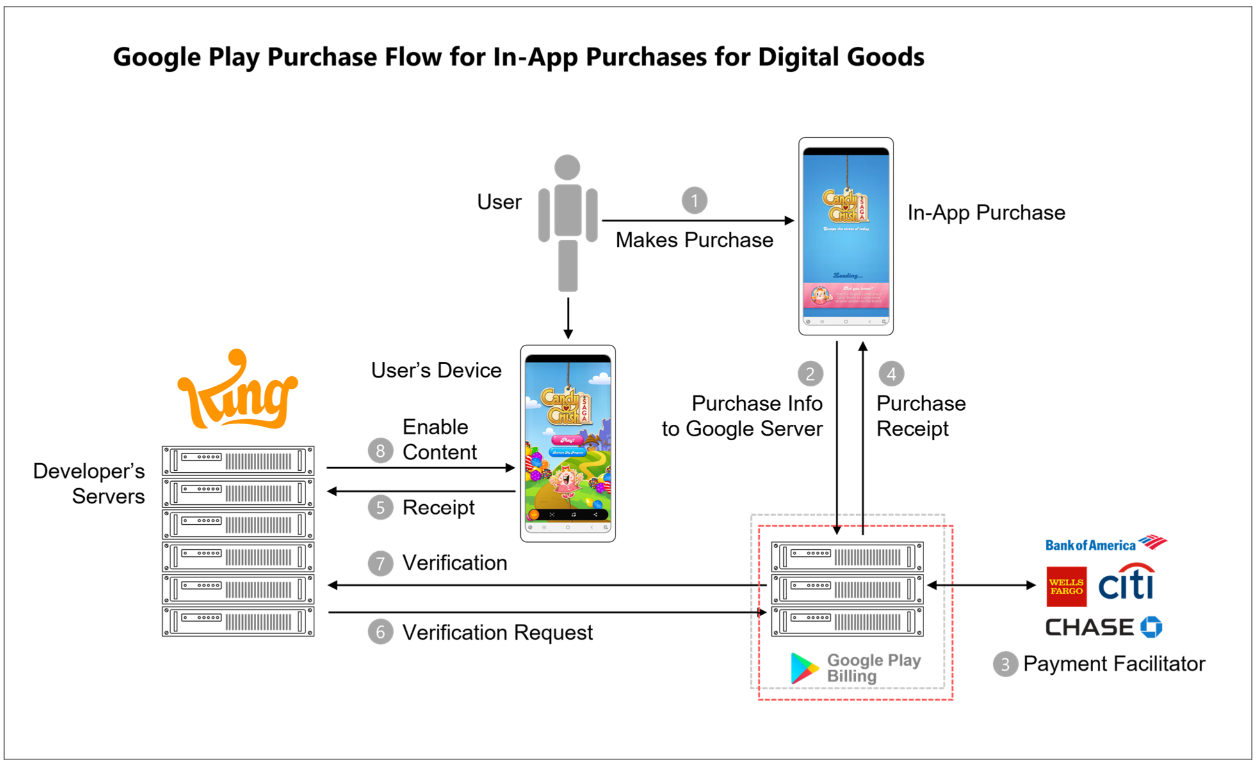
4. Order fulfillment where the merchant ships, transmits, or otherwise supplies the inventory allotment to the customer.

An in-app purchase is not complete until the customer is able to access the item they purchased (step 4). This general purchase flow for in-app purchases for digital goods is shown in the following graphic.

General Purchase Flow for In-App Purchases for Digital Goods



For in-app purchases of digital goods from apps downloaded from the Google Play Store, developers are instructed to use Google's proprietary bundle of services that support in-app purchases, which Google refers to as Google Play Billing, as shown in the following graphic. [15] Notably, in an in-app purchase for a digital good, the developer is responsible for the delivery of the digital good purchased. As Google's [L.] Koh testified, in-app item inventory resides on developers' (not Google's) servers. (Deposition of Lawrence Koh, December 9, 2021, at 381:4-382:6) Although by virtue of the use of Google Play Billing in connection with in-app purchases from apps downloaded from the Google Play Store it is Google that provides confirmation of payment,

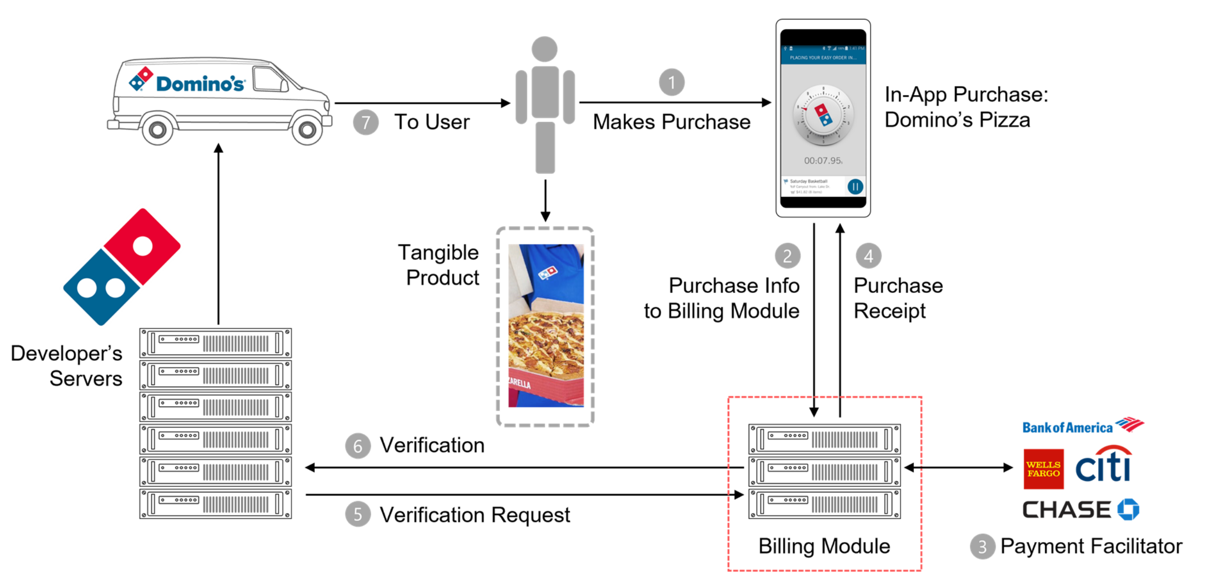


[15] Google currently defines "in-app purchases" requiring the use of Google Play Billing to include: "Play-distributed apps requiring or accepting payment for access to in-app features or services, including any app functionality, digital content or goods," subject to specified exceptions, available at https://support.google.com/googleplay/android-developer/answer/9858738.

it is the developer, not Google, that releases and delivers in-app content to the consumer. (*Id.* at 383:3-21).

The key distinction between an in-app purchase of a physical good and a digital good is the actual fulfillment of the purchase. The following illustration of an in-app purchase of physical goods shows that it is substantially the same as an in-app transaction for digital goods. The key part of the transaction that is impacted is the actual fulfillment of the order – a function that Google does not perform. A physical good must be physically delivered to the purchaser, such as a ride purchased through Uber or a piece of furniture purchased through Amazon's shopping app, while a digital good can simply be added to an account or other inventory tracking system. In either case, the developer or app owner is ensuring fulfilment.

General Purchase Flow for In-App Purchases for Physical Goods



IV. GOOGLE HAS CREATED TECHNICAL BARRIERS FOR NON-GOOGLE APP STORES

Non-Google app stores that are not pre-installed on mobile devices face several technical barriers on Google Android devices. These technical barriers make it harder to install and to use non-Google app stores on Google Android as compared to the Google Play Store.

A. Google Android Forces Users To Go Through the Unknown Sources Flow To Install Apps, Including Non-Google App Stores, Outside Of The Google Play Store

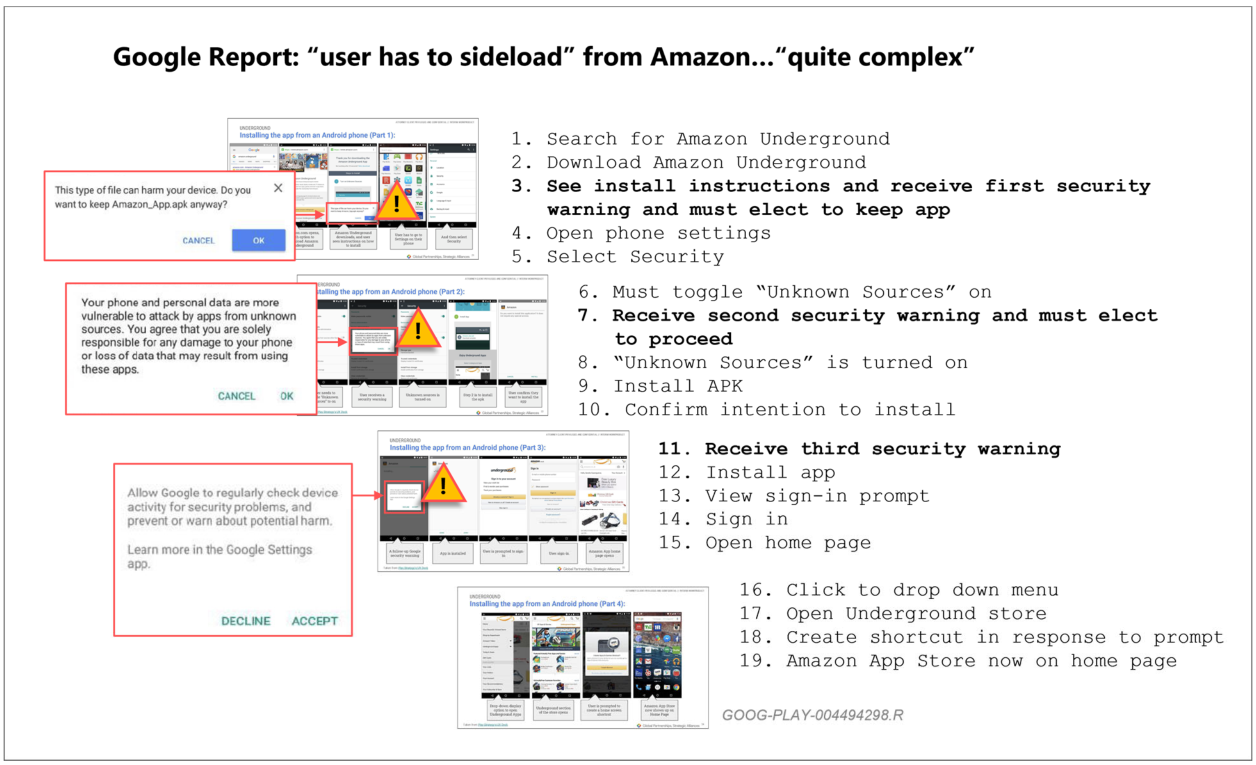
As I explained above, installation of apps by users outside of Google Play triggers the extensive friction of the Unknown Sources Flow. In Google Android, only apps, including app stores, that are pre-installed on mobile devices can have the INSTALL_PACKAGES permission and no third-party apps (i.e., apps that are not developed by Google or OEMs) may be granted the INSTALL_PACKAGES permission.

It follows that any app, including an app store, that is installed through the Unknown Sources Flow will not have the INSTALL_PACKAGES permission. As a result, a user will trigger the Unknown Sources Flow any time he or she attempts to install (1) a non-Google app store or (2) the first app he or she downloads from a non-Google app store that was not pre-installed on his or her device. It is significant that the only app store pre-installed on *all* Google Android devices is the Google Play Store. Thus, the only app store that possesses the INSTALL_PACKAGES permission on *all* Google Android devices is Google Play Store.

As a default, Google Android blocks users from sideloading apps, including app stores, because they do not have INSTALL_PACKAGES permission. To get around that default block, users must go through the Unknown Sources Flow for each source they wish to use to install other apps. For example, even where the user has identified the developer and has gone directly to developer's website through a browser, when the user attempts to download the developer's

app directly to the user's device, the Unknown Sources Flow is triggered. The Unknown Sources Flow has been a part of AOSP Android and Google Android since the initial versions. However, the first iteration of the flow offered much less resistance to users wishing to install apps from outside of the Play Store as it only required the user to change a global setting, rather than a setting per source app.

Over time, Google has modified both the warnings displayed to users and the process through which users authorize the installation of non-Play Store apps. For instance, in March 2016 (Google Android version 6), there were 19 steps for sideloading the Amazon app store (then called "Amazon Underground"). GOOG-PLAY-004494298.R at 318.R to 321.R.[16]



[16] GOOG-PLAY-000575018.R at 019.R-022.R shows that these same steps were in effect in November 2015.

As seen in the foregoing illustration, in this version of the Unknown Sources Flow, users had to navigate through the following three warnings.

1) This type of file can harm your device. Do you want to keep Amazon_App.apk anyway?

2) Your phone and personal data are more vulnerable to attack by apps from unknown sources. You agree that you are solely responsible for any damage to your phone or loss of data that may result from using these apps.

3) Allow Google to regularly check device activity for security problems and prevent or warn about potential harm.

However, after going through the Unknown Sources Flow to download the Amazon app store, users did not encounter the flow again when downloading an app from the Amazon app store.

However, a user must go through the Unknown Sources Flow when he or she downloads an app store and when he or she downloads the first app from the app store due to a change made by Google. Prior to September 2018, when a user identified and changed the permission toggle so that a user could download an app from an "unknown source" to his or her device, the permission toggle remained in that setting. In September 2018, however, Google released Google Android O (also known as "Oreo"). From that point forward, a user had to change the permission toggle every time an installation was attempted from a different "unknown source" (*e.g.*, browser, file explorer, third-party app stores, etc.). In other words, when a non-Google app store is sideloaded from a browser, first the browser must be toggled on as an "unknown source" from which an app can be downloaded. After, the app store itself is installed, it must be toggled on as an "unknown source" from which apps can be downloaded. Even after the setting has been changed for a given source, the user is still presented with a "speed bump" alert each time they wish to install an app. Google calls this a "Per Source Installation Consent dialog" (PSIC).

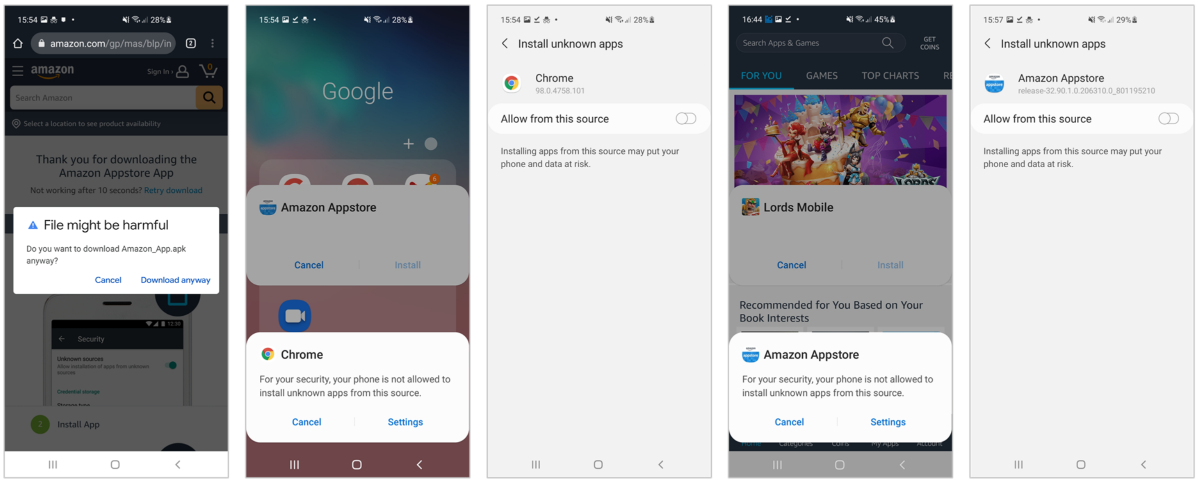
Google acknowledged that this "fundamentally changed the way unknown sources works" and that it resulted in "[m]uch more friction." GOOG-PLAY-000219435.R at 447.R-452.R.[17]

Thus, in Google Android version 11 (released September 2020), if a user wishes to download an app store, such as the Amazon app store, outside of Google Play, the user encounters an Unknown Sources Flow similar to the Google Android version 6 Unknown Sources Flow. Unlike Google Android version 6, however, when the user downloads his or her first app from the sideloaded app store in Google Android version 11, he or she encounters the Unknown Sources Flow again.

Moreover, in Google Android version 11, to download the third-party app store and the first app from that third-party app store, the user must navigate through the following five warnings.

[17] Though it remains largely the same, Google has made some limited improvements to the Unknown Sources Flow in Android 12. For example, in Android 12, the system automatically prompts the user to install the previously requested application immediately following the permission toggle, rather than forcing the user to navigate back to the app store that requested the installation.

Warnings encountered in downloading third-party app store and first app from third-party app store (Google Android version 11)

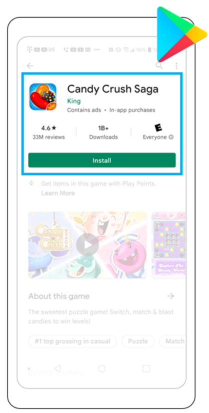


In contrast to the friction a user encounters when he or she wants to install and use a non-Google app store, a user *never* has to go through the sideloading friction of the Unknown Sources Flow to install the Google Play Store because the Google Play Store is installed on *every* Google Android device. In addition, a user *never* has to go through the sideloading friction of the Unknown Sources Flow to install an app from the Google Play Store. Instead, the installation of an app from the Google Play Store happens with one click.

Downloads From Google Play Are Quick and Easy

No Unknown Sources Flow
(Google Play)

Choose
↓
Install



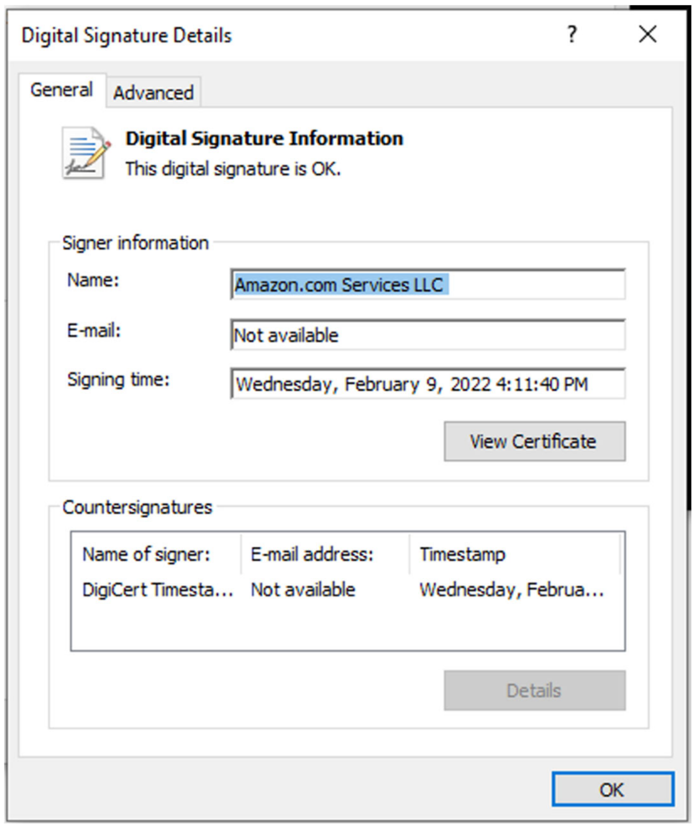
User clicks Google Play
(icon on home screen)

1. **Selects Game or App**
2. **Clicks Install**

The Unknown Sources Flow is the primary mechanism that users can apply to install apps on Google Android devices outside of OEM pre-loaded app stores. (GOOG-PLAY 001317740 at 747; GOOG-PLAY-001317740 at 744; GOOG-PLAY-004494298.R at 321.R; GOOG-PLAY-004704453.R at 495.R; GOOG-PLAY-004704453.R at 498.R). The Unknown Sources Flow mechanism is triggered by apps installed from web downloads, including non-Google app stores. It is also triggered by apps installed by a user who installs from non-Google app stores that are not pre-loaded on his or her device.

The Unknown Sources Flow friction designates well-known and respected apps, such as Amazon, as "unknown sources" that are potentially harmful to a user's phone. Likewise, Google has used it to declare that updates from sources other than the Google Play Store are categorically "unsafe." GOOG-PLAY-000046830.R at 848.R. This designation is applied even though these sources are readily identifiable and reputable, and a plethora of mechanisms exist to easily verify not only the identity of the developer, but to also validate the authenticity and safety

of the app. For instance, the Amazon Music App on Windows is signed by a certificate clearly identifying Amazon as the publisher. This certificate not only identifies the developer in a verifiable manner, but it also demonstrates that the app has not been tampered with since the developer published the app:



Amazon is not only an operator of a significant portion of the internet's infrastructure but it also has manufactured its own line of Android devices. *See, e.g.,* Deposition of David Kleidermacher, February 4, 2022, at 451:17-25 (Amazon "may be reasonably well regarded. It's being used on a very large number of Android devices, like the Fire devices, Fire tablets and whatnot"). However, Google Android still presents the user with security warnings when he or she attempts to install Amazon's app store or attempts to install an app from Amazon's app store. But Google's VP of Engineering for the Google Play Store has acknowledged that not all "unknown sources" make a user's phone and personal data more vulnerable. *Id.* at 420:25-421:8

(Q: "Does that risk apply to all of what qualifies here as unknown sources or just to some unknown sources?" A: "It would apply to any – it wouldn't be all."). Moreover, he suggested that Google has the capability of identifying apps. *See, e.g., Id.* at 457:14-458:4 ([redacted]); *Id.* at 409:4-412:23 ([redacted]); *Id.* at 390:13-19 ([redacted]); *Id.* at 407:18-408:8 [redacted]).

The Unknown Sources Flow creates a significant deterrent to sideloading apps. In a 2020 slide deck, Epic estimated that even among users who tried to sideload Epic apps from the Epic website, the Unknown Sources Flow resulted in 37% of users failing to complete installation. EPIC_GOOGLE_03982547 at 563. One Epic executive concluded in 2019 that "[redacted]" EPIC_GOOGLE_01761929. Moreover, a global Google analysis of the Unknown Sources Flow in 2016 showed that in versions of Android prior to version "O," "62% of users who encountered the 'unknown source' warning proceeded to enable the setting," meaning that nearly 40% of users halted an installation attempt on one of the very first steps in the Unknown Sources Flow. GOOG-PLAY-004904016.R at 4038.R. As far back as 2016, Google recognized that the Unknown Sources Flow meant that downloading the Amazon store was "still quite complex" because it was "NOT available in Play Store, so a user has to sideload this app." GOOG-PLAY-004494298.R at 317.R.

As previously discussed, there are limited avenues for installation of third-party apps outside of pre-loaded app stores. The Unknown Sources Flow appears designed to discourage users from installing apps outside the Google Play Store. GOOG-PLAY-000219435.R at 457.R ("[A]dding any kind of friction will reduce installs (proposition is that there is about a [redacted] percent dropoff for every acquisition in Play, so for non-Play we expect similar)"); Deposition of David Kleidermacher, February 4, 2022, at 454:6-17 ("Q: Holding all else equal, introducing friction in a UI [user interface] will deter some downloaded and installations, correct? A: We saw some metrics earlier that made that claim. I think it's logical that, like, the more difficulty a UI is to navigate, the more likely a user would be, like, canceling out of the UI."); Deposition of Tian Lim, December 2, 2021, at 258:25-259:8 ("Q. What does adding friction do to the likelihood of a user conversion for a developer? A. . . . I think more steps typically lead to lower conversion.").

Moreover, Google views the Unknown Sources Flow as "pain points" for other app stores. GOOG-PLAY-004904016.R at 4033.R (referring to the unknown sources steps as "3P store pain points"). In 2017, Google noted that one way to encourage users to stay on the Google Play Store rather than use the Amazon app store was to "create more third party apk install friction, e.g., speed bump type hurdles." GOOG-PLAY-000297605.R at 634.R. In 2018, Google determined the Amazon store had limited success with users despite "strong capabilities" because of "friction from side-loading." GOOG-PLAY-000005203.R at 220.R. Moreover, Google noted in 2019 that both Amazon and Microsoft would have to overcome developer "increased friction for discovery/installation" and user "complexity of alternate discovery – friction (unknown sources) if not preinstalled" to launch game distribution platforms to compete with the Google Play Store. GOOG-PLAY-002011285.R at 288.R.

In April 2020, Google stated that it was "driving users towards denying app installer permissions, which allow users to install apps from other apps" (GOOG-PLAY-002910052.R at 056.R). Moreover, the change to "per source" permission that Google made in the Oreo version of Google Android in 2018 and that I discuss above was meant to "[r]emind users that they have given apps permissions to install other apps and drive them towards Android settings for them to revoke those permissions." GOOG-PLAY-002910052.R at 057.R-058.R. While Google said its goal was to "[r]educe the installation of apps from unknown sources as they are 8 time (sic) more likely to contain malware" (GOOG-PLAY-002910052.R at 056.R), Google elsewhere acknowledged that "[a]pp stores generally have relatively low malware install rates" including stores like "Café Bazaar (0.15%), F-Droid (0.05%), Epic Store (0%), Amazon (0.7%)" GOOG-PLAY-004268238.R at 242.R.

B. Google Android Forces Users To Manually Update Apps Downloaded Outside Of The Google Play Store

As I explained above, in Google Android version 11 and in all previous versions, the INSTALL_PACKAGES permission is also required to update apps automatically without user action. If an app, including an app store, does not have INSTALL_PACKAGES permission on devices running those versions of Google Android, an app can only be updated by the user manually and while the user has the app store open. So even if a user goes through the friction of the Unknown Sources Flow to sideload a non-Google app store, and even if a user goes through the friction of the Unknown Sources Flow to install an app from that non-Google app store, the user must open the app store and respond to the "speed bump" prompt to manually install the update to that app.

This speed bump potentially increases security risks posed to users because developers are unable to automatically update their apps to patch security holes. Instead, they must first

notify users of a potential problem and then hope that users navigate the manual process to install the update. As I mentioned in footnote 14 above, with Google Android version 12 (released October 2021), Google began allowing users to manually opt-in to automatic updates by third-party app stores. This new opt-in flow appears even though users must still go through Google's Unknown Sources Flow to sideload third-party app stores, including reputable third-party app stores like Amazon.

In contrast, because the Google Play Store always has INSTALL_PACKAGES permission, a user *never* has to go through the manual process to update an app he or she has downloaded from the Google Play Store or opt-in to allow the Google Play Store to automatically update apps; updating happens automatically, in the background, typically without the user ever knowing. This feature makes the Google Play Store updating process much more user friendly than the degraded experience users must endure to manually update an app downloaded from a user-installed non-Google app store.

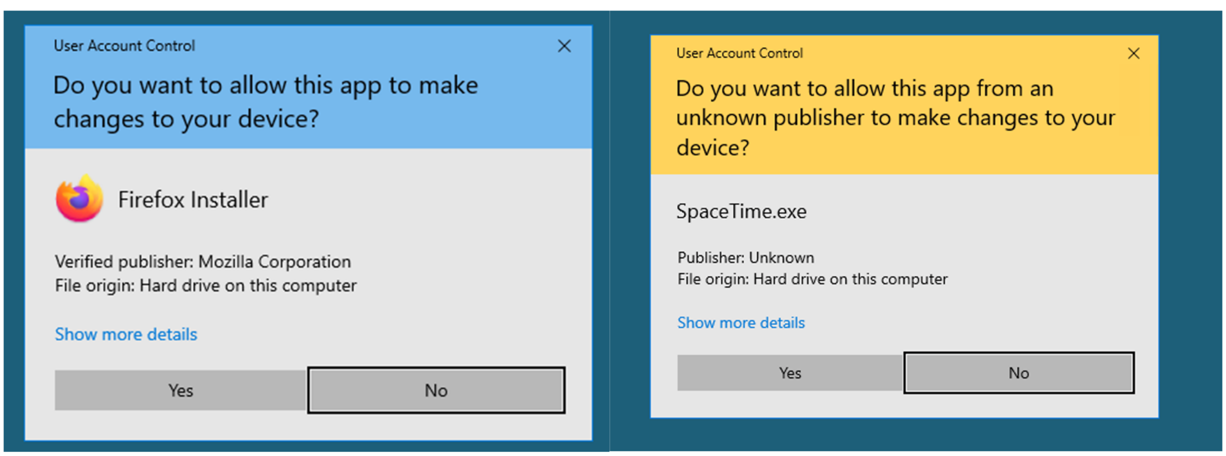
C. Google Could Reasonably Allow Non-Google App Stores To Bypass The Unknown Sources Flow Without Compromising Security

Google could implement a system that allowed it to identify trustworthy developers and allow their apps to bypass the Unknown Sources Flow without compromising security. This process is known as "whitelisting."

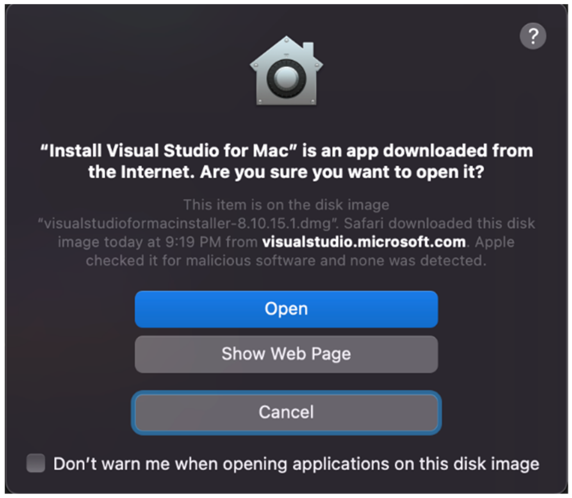
As an initial matter, Google can identify the source of an app. GOOG-PLAY-000398862 at 8915 to 8916 (explaining that Google can identify apps from "reputable" companies). The testimony of Mr. Kleidermacher, Google's VP of Engineering for the Play Store, confirmed this capability. *See, e.g.,* Deposition of David Kleidermacher, February 4, 2022, at 457:14-458:4 ([redacted]); *Id.* at 390:13-19 ([redacted]

[redacted]); *Id.* at 407:18-408:8 ([redacted]. And Mr. Kleidermacher also confirmed that once an app is identified as trustworthy, Google can allow it to bypass the Unknown Sources Flow. *Id.* at 409:4-412:23 ([redacted]).

Moreover, other systems have implemented whitelisting processes to allow users to install apps from developers that have been vetted and display messages that indicate the source of the app is known and verified. For example, Microsoft[18] and Apple both provide similar attestation processes that allow developers to obtain a signed version of their app. Installing an app that has the appropriate signature displays different prompts than when an untrusted app is installed:



[18] https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works



In both instances, the verification of the publisher is achieved through the use of a digital signature that allows the user to cryptographically verify the identity of the publisher and allows the system to display more accurate prompts to the user. These signatures allow the user's computer to:

1. display the publisher's name to the user;
2. indicate that the publisher has gone through the necessary verification steps with either Apple or Microsoft; and
3. indicate that the app has not been modified since the developer published it.

The use of digital signatures to verify the authenticity of a particular package is not a novel concept, and in fact is already used by Google Android for different parts of the app distribution process[19] as well as many other systems to provide security to users.

Microsoft and Apple can set the requirements for these programs, can verify and confirm that the necessary credentials and processes are in place, and can manage this process at a significant scale. In fact, for Apple, *all* developers must go through a verification and validation

[19] https://developer.android.com/studio/publish/app-signing

process to satisfy the Gatekeeper system within iOS. So not only is this whitelisting process possible and easy to implement, but it is also currently implemented in the real world by at least two major platform vendors.

Another mechanism for whitelisting apps would be to use the signature of the app itself. Google could implement a system where apps were submitted to Google and verified. That same, unmodified app package could then be distributed outside of the Play Store with Google verifying that the app was previously scanned.

Google also has the capability to whitelist specific app *stores* based on security information reviewed by either Google itself or by third party certification programs. David Kleidermacher, Google's VP of Engineering for the Play Store, considered the possibility of creating a "whitelist of higher assurance app stores." GOOG-PLAY-004506533 at 533 (authored by Mr. Kleidermacher). As he later explained, "The concept that I was thinking about here is if we had a way of determining relative confidence in app stores, if we had such a mechanism or way of doing that, then we could treat that -- we could treat them differently from, say, the full universe of all app stores and sources. . . . It's a topic that has come up from time to time with a variety of people." Deposition of David Kleidermacher, February 4, 2022, at 447:6-20.

Amazon apparently tried to share its own safety protocols with Google as early as 2017, to remove the warnings to users when attempting to install the Amazon app store. Google, however, refused. GOOG-PLAY-004722290 at 291 ("To address our joint effort to improve security on the Android platform, Amazon asked us to whitelist the new Amazon apps and games store to bypass unknown sources. They offered to share documentation of their security and approval processes of 3P apps to ensure they would be compliant. . . . Given our view on security overall policy approach, this proposal is a non-starter and not something we would

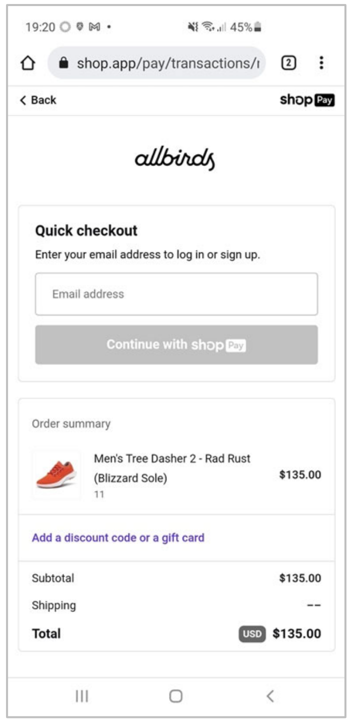
support."). See also, Deposition of David Kleidermacher, February 4, 2022, at 452:16-18 (noting Amazon "may have metrics of things that they see that might be useful").

Mr. Kleidermacher has also considered the prospect of "independent certification" by "some independent entity that is trusted by the general mobile world" to create the whitelist of "higher assurance app stores." Deposition of David Kleidermacher, February 4, 2022, at 448:23-449:17: ("And so if a standard could be created that would be sufficiently robust and effective, that should actually be able to have a rigorous and -- a rigorous form of evaluation of store safety and quality, and that could be independently or certified by some entity -- some independent entity that is trusted by the general mobile world, that is one concept that I have thought about from time to time.") I am unaware of Google publicly pursuing such certification programs, despite acknowledging that "there [are] plenty of entities out there that run certification programs." *Id.* at 449:25-450:12.

V. THERE IS NO TECHNICAL REASON TO REQUIRE THE USE OF GOOGLE PLAY BILLING FOR IN-APP TRANSACTIONS OF DIGITAL GOODS

Google inserts itself into the in-app transaction flow through the requirement that developers use Google Play Billing. Google's insertion of itself in the in-app transaction flow is technologically unnecessary. After a user has downloaded an app from the Google Play Store, neither Google nor its Play Store are necessary, as a matter of technological efficiency, to facilitate the purchase and distribution of in-app digital content between the developer and the consumer.

Use of Google Play Billing associated with in-app transactions can easily be separated from the Google Play Store. For example, Google's in-app purchase user interface displays the purchase information to the user, displaying a graphic, the price, and other information. This interface is identical to the experience provided by the Shop Pay interface in the mobile browser:



Google recognizes that Google Play Billing is functionally a separate product. GOOG-PLAY-001088593; GOOG-PLAY-006997722.C at 725 (Google proposal to decouple Play from Google Play Billing); GOOG-PLAY-001088593 ([redacted]).

Moreover, companies have offered products and services for sale via the internet in a secure manner for decades without any connection to an app store. There are numerous companies and services that exist to facilitate the secure capture of payment information to purchase both physical and digital goods, including PayPal,[20] Braintree,[21] Stripe,[22]

[20] https://www.paypal.com/us/webapps/mpp/paypal-checkout

[21] https://www.braintreepayments.com

[22] https://stripe.com

Authorize.Net,[23] Shopify,[24] and others. While Google Play Billing does allow for users to save a payment method, so would any other payment network (*e.g.*, Shop Pay,[25] Pay with Amazon,[26] etc.). Moreover, the Payment Card Industry (PCI) regulates and standardizes the security of card data and transactions through their Data Security Standards (PCI-DSS). All merchants and service providers wishing to handle card transactions are required by the card networks to comply with these standards. All merchants and service providers wishing to handle card transactions are required to comply with these standards by the card networks.

Allowing developers to integrate with any of these payment service methods, rather than Google Play Billing, was a reasonable and viable option for Google. (GOOG-PLAY-005649820, -007271749, -007271750, -005653611, -005653612.R, -001076451, -001076452.R, 005659061, -004284846, -005563726) Indeed, Google has recently announced a partnership with Stripe via Google's Firebase SDK for the purchase of physical goods,[27] further implying Google's acknowledgement of Stripe's payment security and capability for Android in-app purchases.

Depending on the level of investment that developers wish to make in their payment infrastructure, they can choose to handle more or less of the process internally. However, in the most straightforward setup, which is used by most merchants on the Internet and requires the least amount of investment, the payment system, such as Stripe or Authorize.net, handles steps 1 and 2 of the general in-app purchase steps introduced in Part III G and provides a notification to the merchant when payment is successful. The merchant is then able to handle fulfillment of the

[23] https://www.authorize.net

[24] https://www.shopify.com

[25] https://www.shopify.com/partners/blog/shopify-android-buy-sdk

[26] https://paymentservices.amazon.com/docs/EN/23c.html

[27] https://firebase.google.com/products/extensions/stripe-firestore-stripe-payments

order. In fact, this system is used currently to distribute both physical and digital goods outside of a mobile environment.

For example, the clothing retailer AllBirds uses ShopPay to process payment information on their website and the software vendor XK72 sells its network analysis software called Charles Proxy using a payment system called Paddle. These payment systems, among others, already have mobile friendly SDKs which could be used in apps on Google Android devices.[28] [29] [30] [31] By inserting Google Play Billing as the sole payment system for in-app purchases of digital products, Google deprives developers of the opportunity to choose among other competing services.

VI. CONCLUSION

For the reasons stated above, I conclude that (1) Google has imposed technological restraints that artificially restrict or inhibit consumer access to competitive app stores and to apps sold through competitive app stores or other distribution channels, and (2) there is no technical reason for Google to include its billing service product, Google Play Billing, in the flow of purchases made directly in apps that users have downloaded to their mobile device.

Douglas Craig Schmidt, Ph.D.:

____

Executed on February 28, 2022

---

[28] https://paddle.com/platform/in-app-purchase

[29] https://stripe.dev/stripe-android/

[30] https://github.com/paypal/android-checkout-sdk

[31] https://developer.authorize.net/api/reference/features/in-app.html

EXHIBIT A

Dr. Douglas Craig Schmidt

Cornelius Vanderbilt Professor of Engineering
Department of Electrical Engineering & Computer Science
Vanderbilt University
Nashville, TN 37203

douglas.c.schmidt@vanderbilt.edu
(TEL) 615-294-9573
(FAX) 615-343-7440
(WEB) www.dre.vanderbilt.edu/∼schmidt/

Educational Background

- **Ph.D. Computer Science**, summer 1994, University of California, Irvine
  Dissertation: "An Object-Oriented Framework for Experimenting with Alternative Process Architectures for Parallelizing Communication Subsystems."
  Co-advisors: Dr. Tatsuya Suda and Dr. Richard W. Selby.
- **M.S. Computer Science**, summer 1990, University of California, Irvine, specializing in software engineering.
- **M.A. Sociology**, summer 1986, College of William and Mary, Williamsburg, Virginia
  Thesis: "A Statistical Analysis of University Resource Allocation Policies."
  Advisor: Dr. Michael A. Faia.
- **B.A. Sociology**, summer 1984, College of William and Mary, Williamsburg, Virginia.

Professional Experience

1. **7/1/18 – present: Associate Provost of Research Development and Technologies**
   Develop cohesive and sustainable information technology (IT) services to advance research and scholarship across Vanderbilt's ten schools and colleges; develop scalable storage and processing solutions by leveraging on-campus and cloud data storage services, as well as creating big data research cores and core-related services; and implement NIST 800-171 compliant IT services.
2. **8/1/18 – present: Co-Director of the Vanderbilt Data Science Institute**
   Facilitate highly innovative research and education initiatives that build on Vanderbilt University's current strengths, promote new collaborations, and establish a cohesive institutional framework that embraces Vanderbilt's diverse campus, while establishing the university as a leader in data science research and education.
3. **2/17 – present: Cornelius Vanderbilt Professor of Engineering**
   Received an endowed chair in recognition of my scholarship, intellect, and leadership in the field of computer science and computer engineering.
4. **1/03 – present: Full Professor with tenure**
   Conducting research on patterns, optimizations, and experimental analysis of advanced generative software techniques that facilitate the development of distributed real-time and embedded middleware and model driven architectures running over high-speed networks and interconnects in the Department of Electrical Engineering and Computer Science at Vanderbilt University.
5. **02/16 – 7/31/18: Associate Chair of Electrical Engineering and Computer Science**
   Provide intellectual leadership within the EECS department. Coordinate with EECS Chair to assist in EE, CS, and CompE curriculum development and course staffing. Assist the faculty in building industry and federal programs for EECS. Assist the Chair in mentoring junior EECS faculty. Assist the EECS Chair in improving the ranking of the EECS programs. Assist the Chair in increasing the quality and number of undergraduate and graduate student applications to the EECS programs.
6. **12/04 – 1/16: Associate Chair of Computer Science and Engineering**
   Provide intellectual leadership within the CS program. Coordinate with EECS Chair to assist in CS and CompE (CS&E) curriculum development and course staffing. Assist the faculty in building industry and federal programs centered in CS&E and IT for EECS. Assist the Chair in mentoring

junior CS&E faculty. Assist the EECS Chair in improving the ranking of the CS&E programs. Assist the Chair in increasing the quality and number of undergraduate and graduate student applications to the CS&E programs.

7. **4/13 – 2/18: Member of the Board of Directors at Real-Time Innovations (RTI)**.
Work with the CEO and other members of the Board of Directors of RTI to help assess company technical and business strategy.

8. **1/12 – present: Visiting Scientist at the Software Engineering Institute**
Assist the SEI Director's Office in formulating the SEI's technology strategy for R&D projects and external relationships by aligning the expertise of the SEI technical staff to identify and respond to the needs of sponsors, customers, and partners and help the SEI shape future innovations in complex software-reliant systems.

9. **7/11 – 7/13: Adjunct Professor of Software Engineering** in the Institute for Software Research in the School of Computer Science at Carnegie Mellon University.

10. **9/10 – 12/11: Deputy Director and Chief Technology Officer at the Software Engineering Institute (SEI)**
Lead the formulation of the SEI's technology strategy for R&D projects and external relationships by aligning the expertise of the SEI technical staff to identify and respond to the needs of sponsors, customers, and partners and help the SEI shape future innovations in complex software-reliant systems.

11. **07/05 – 8/10: Visiting Scientist at the Software Engineering Institute**
Assisted Linda Northrop and the Ultra-Large-Scale (ULS) Systems team to define the challenge problems, promising technology areas, and research roadmaps for the national R&D effort on building the software-reliant systems of the future that are likely to have billions of lines of code. This activity is defining a broad, multi-disciplinary research agenda for developing ULS systems of the future.

12. **06/09 – 8/10: Chief Technology Officer for Zircon Computing**
Assisted in the strategic direction of Zircon Computing technology development in the areas of adaptive distributed computing middleware for high-performance and real-time applications. Help to formulate the technology strategy for open-source middleware platforms, R&D partnerships, and external relationships.

13. **6/07 – 8/07: Visiting Professor at Trinity College Dublin**
Worked with Professor Vinny Cahill and the Distributed Systems Group at Trinity College on topics pertaining to service-oriented architectures and autonomic computing.

14. **10/06 – 5/09: Chief Technology Officer for PrismTechnologies**
Assisted in the strategic direction of PrismTechnologies technology development in the areas of open-source middleware platforms and model-driven tools. Help to formulate the technology strategy for open-source middleware platforms and model-driven tools, R&D partnerships, and external relationships.

15. **3/02 – 12/02: Program Manager**
Led the National effort on middleware as a Program Manager for over $60 million dollars of funding at the DARPA Information Exploitation Office (IXO). Programs include Program Composition for Embedded Systems (PCES) and National Experimentation Platform for Hybrid and Embedded Systems (NEPHEST).

16. **9/01 – 3/02: Deputy Director**
Served as the Deputy Director for the DARPA Information Technology Office (ITO), helping set and guide the National IT research and development agenda and manage programs on autonomous systems, network-centric command and control systems, combat systems, real-time avionics systems, distributed real-time and embedded systems, and augmented cognition for the U.S. Department of Defense.

17. **6/00 – 3/02: Program Manager**
Led the National effort on middleware as a Program Manager for over $60 million dollars of funding at the DARPA Information Technology Office (ITO). Programs included the Program Composition for Embedded Systems (PCES).

18. **6/01 – 6/02: Co-chair for the Software Design and Productivity (SDP) Coordinating Group**
The SDP Coordinating Group formulates the multi-agency research agenda in fundamental software design for the Federal government's Networking and Information Technology Research and Development (NITR&D) Program, which is the collaborative IT research effort of the major Federal science and technology agencies.
19. **8/99 – 2002: Associate Professor with tenure**
Conducted research on patterns, implementation, and experimental analysis of object-oriented techniques that facilitate the development of high-performance, distributed real-time and embedded computing systems on parallel processing platforms running over high-speed networks and embedded system interconnects in the Department of Computer Engineering at the University of California, Irvine.
20. **6/99 – 8/99: Associate Professor with tenure**
Conducted research on patterns, implementation, and experimental analysis of object-oriented techniques that facilitate the development of high-performance, distributed real-time and embedded computing systems on parallel processing platforms running over high-speed networks and embedded system interconnects in the Department of Computer Science and the Department of Radiology at Washington University in St. Louis.
21. **6/98 – 6/99: Associate Professor without tenure (early promotion)**
Conducted research on patterns, implementation, and experimental analysis of object-oriented techniques that facilitate the development of high-performance, distributed real-time and embedded computing systems on parallel processing platforms running over high-speed networks and embedded system interconnects in the Department of Computer Science and the Department of Radiology at Washington University in St. Louis.
22. **8/94 – 6/98: Assistant Professor**
Conducted research on object-oriented patterns and techniques for developing highly extensible, high-performance communication frameworks in the Department of Computer Science and the Department of Radiology at Washington University in St. Louis.
23. **3/91 – 8/94: Research Assistant**
Developed object-oriented frameworks for multi-processor-based communication subsystems with Professor Tatsuya Suda at the University of California, Irvine.
24. **6/90 – 11/90: Member of the Technical Staff**
Worked as a software engineer for Independence Technologies, which was one of the largest suppliers of enterprise-level TUXEDO systems, providers of professional services, and developers of management and connectivity software to support OLTP environments.
25. **8/88 – 3/91: Research Assistant**
Devised measurement-guided software development techniques for large-scale software systems with Professor Richard Selby at the University of California, Irvine.
26. **6/88 – 8/88: Research Assistant**
Studied the impact of computing on end-users in forty U.S. city governments with Dr. John King and the URBIS project at the Public Policy Research Organization, University of California, Irvine.
27. **Summer of 87: Technical Intern**
Worked with Dr. Peter G. W. Keen at the International Center for Information Technology, Washington D.C. on various projects, including software productivity, videotex, and smartcards.
28. **9/86 – 5/88: Teaching Assistant**
Developed programming assignments, grading tools, and led recitation sessions for a number of undergraduate Computer Science courses at the University of California, Irvine.
29. **Summer of 86: Statistical Programmer**
Programmed SPSS and SAS applications for the "Justice Delayed" project under the direction of Dr. Gene Flango at the National Center for State Courts, Williamsburg, Virginia.
30. **1/85 – 8/86: Research Assistant**
Examined university resource allocation policies via statistical analysis under the direction of Dr. Michael Faia at the College of William and Mary, Williamsburg, Virginia.

Publications

In Print

- **Refereed Journal Publications**

J129 Peng Zhang, Christopher Fonnesbeck, Douglas C. Schmidt, Jules White, Samantha Kleinberg, Shelagh A. Mulvaney, "Understanding Barriers to Self-Management in Type 1 Diabetes Using Machine Learning and Momentary Assessment," the JMIR *Journal of mHealth and uHealth*, 2022 (to appear).

J128 Summer Weber, Elyse Shearer, Shelagh Mulvaney, Douglas C. Schmidt, Chris Thompson, Jessica Jones, Haseeb Ahmad, Martina Coe, and Pam Hull, "Prioritization of Features for Mobile Phone Applications for Families in a Federal Nutrition Program for Low-income Women, Infants, and Children: User-Centered Design Approach," *JMIR Formative Research*, Vol 5., No 7., July 2021.

J127 Alex Roehrs, Cristiano A. da Costa, Rodrigo R. Righi, Andre H. Mayer, Valter F. da Silva, Jose R. Goldim, and Douglas C. Schmidt, "Integrating Multiple Blockchains to Support Distributed Personal Health Records," the SAGE *Health Informatics Journal*, April, 2021.

J126 Zhongwei Teng, Peng Zhang, Xiao Li, William Nock, Denis Gilmore, Marcelino Rodriguez-Cancio, Jules White, Jonathan C. Nesbitt, Douglas C. Schmidt, "Authentication and Integration Approaches for mHealth Apps from a Usability View," the journal *Advances in Electrical and Electronic Engineering*, North America, 19, March, 2021.

J125 Scott Eisele, Aron Laszka, Douglas C. Schmidt, and Abhishek Dubey, "The Role of Blockchains in Multi-Stakeholder Transactive Energy Systems," the journal *Frontiers in Blockchain: Emerging Technologies and Blockchain in Action: Applications in Supply Chain Management and Energy*, to appear 2021.

J124 Peng Zhang, Chris Downs, Nguyen Thanh Uyen Le, Cory Martin, Paul Shoemaker, Clay Wittwer, Luke Mills, Liam Kelly, Stuart Lackey, Douglas C. Schmidt, Jules White, "Towards Patient-centered Stewardship of Research Data and Research Participant Recruitment with Blockchain Technology," the *Frontiers in Blockchain special selection on Non-Financial Blockchain*, 2020, volume 3, pps. 1-32.

J123 Yao Pan, Fangzhou Sun, Jules White, Douglas C. Schmidt, Jacob Staples, Lee Krause, and Zhongwei Teng, "Detecting Web Attacks with End-to-End Deep Learning," the Springer *Journal of Internet Services and Applications*, 2019, volume 10, number 16, pps. 1-22.

J122 Shelagh Mulvaney, Lori Laffel, Korey Hood, Cindy Lybarger, Sarah Vaala, and Douglas C. Schmidt, "A Mobile App Identifies Momentary Psychosocial and Contextual Factors Related to Mealtime Self-Management in Adolescents with Type 1 Diabetes," Journal of the American Medical Informatics Association, Oxford University Press, 2019, Volume 26, Number 12, pps. 1627-1631.

J121 Maria E. Powell, Marcelino Rodriguez Cancio, David Young, William Nock, Beshoy Abdelmessih, Amy Zeller, Irvin Perez Morales, Peng Zhang, C Gaelyn Garrett, Douglas Schmidt, Jules White, and Alexander Gelbard, "Decoding Phonation with Artificial Intelligence (DEP AI): Proof of Concept," the *Laryngoscope Investigative Otolaryngology* journal, Wiley-Blackwell, Volume 4, Issue 3, 2019, pps. 328-334.

J120 Alex Roehrs, Cristiano Andre da Costa, Rodrigo da Rosa Righi, Valter Ferreira da Silva, Jose Roberto Goldim, and Douglas C. Schmidt, "Analyzing the Performance of a Blockchain-based Personal Health Record Implementation," the *Journal of Biomedical Informatics*, Elsevier, volume 92, 2019.

J119 Peng Zhang, Breck Stodghill, Cory Pitt, Cavan Briody, Douglas C. Schmidt, Jules White, Alan Pitt, and Kelly Aldrich, "OpTrak: Tracking Opioid Prescriptions via Distributed Ledger Technology," the *International Journal of Information Systems and Social Change (IJISSC)*, Special Issue On: Blockchain Technology: Platforms, Tools, and Use Cases, IGI Global, Volume 10, Number 2, 2019.

J118 Peng Zhang, Jules White, Douglas C. Schmidt, Gunther Lenz, S. Trent Rosenbloom, "FHIRChain: Applying Blockchain to Securely and Scalably Share Clinical Data," the Elsevier *Computational and Structural Biotechnology Journal* – Blockchain and Distributed Ledger Technologies in Biology, Medicine, and eHealth Special Issue, Volume 16, July 2018, pp 267–278.

J117 Shelagh A Mulvaney, Sarah Vaala, Korey K Hood, Cindy Lybarger, Rachel Carroll, Laura Williams, Douglas C Schmidt, Kevin Johnson, Mary S Dietrich, and Lori Laffel, "Mobile Momentary Assessment and Bio-Behavioral Feedback for Adolescents with Type 1 Diabetes: Feasibility, Engagement Patterns, and Relation with Blood Glucose Monitoring," ¡EM¿Journal of Diabetes Technology and Therapeutics¡/EM¿, Vol 20, No. 7, July 2018, pp 465–474.

J116 Subhav Pradhan, Abhishek Dubey, Shweta Khare, Saideep Nannapaneni, Aniruddha Gokhale, Sankaran Mahadevan, Douglas C Schmidt, Martin Lehofer, "CHARIOT: A Holistic, Goal Driven Orchestration Solution for Resilient IoT Applications," the ACM Transactions on Cyber-Physical Systems, Vol 2, No. 3, July 2018, pp 1-37.

J115 Hull PC, Emerson JS, Quirk ME, Canedo JR, Jones JL, Vylegzhanina V, Schmidt D, Mulvaney S, Beech B, Husaini BH, "A Smartphone App for Families With Preschool-Aged Children in a Public Nutrition Program: Prototype Development and Beta-Testing," *Journal of Medical Internet Research (JMIR): mHealth and uHealth*, Vol 5, No. 8, August, 2017, pp 1–19.

J114 Yao Pan, Jules White, Douglas C. Schmidt, Ahmed Elhabashy, Logan Sturm, Jaime Camelio, and Christopher Williams, "Taxonomies for Reasoning About Cyber-physical Attacks in IoT-based Manufacturing Systems," Special Issue on Advances and Applications in the Internet of Things, edited by Vicente Garcia Diaz, International Journal of Interactive Multimedia and Artificial Intelligence, volume 4, number 3, 2017, pp. 45-54.

J113 Gordon Blair, Douglas C. Schmidt, and Chantal Taconet, "Middleware for Internet Distribution in the Context of Cloud Computing and the Internet of Things," Springer Journal Annals of Telecommunications, April 2016, Volume 71, Issue 3, pp. 87-92.

J112 Yu Sun, Jules White, Sean Eade, and Douglas C. Schmidt, "ROAR: A QoS-Oriented Modeling Framework for Automated Cloud Resource Allocation and Optimization", the Journal of Systems and Software, Elsevier, volume 116, issue C, June 2016 pp. 146.161.

J111 Nick Guertin, Brian Womble, Paul Bruhns, Douglas C. Schmidt, Adam Porter, and Bill Antypas, "Management Strategies for Software Infrastructure in Large-Scale Cyber-Physical Systems for the US Navy," Cutter IT Journal, Vol. 28, No. 5, May 2015, pp. 14-18.

J110 Jules White, Josi A. Galindo, Tripti Saxena, Brian Dougherty, David Benavides, Douglas C. Schmidt, "Evolving Feature Model Configurations in Software Product Lines," Journal of Systems and Software, Volume 87, 2014, pp. 119-136.

J109 Akram Hakiri, Aniruddha S. Gokhale, Pascal Berthou, Douglas C. Schmidt, Thierry Gayraud, Software-Defined Networking: Challenges and Research Opportunities for the Future Internet," Journal of Computer Networks, Volume 75, 2014, pp. 453-471.

J108 Hamilton Turner, Brian Dougherty, Jules White, Jonathan Preston, Russell Kegley, Douglas C. Schmidt, and Aniruddha Gokhale, "DRE System Performance Optimization with the SMACK Cache Efficiency Metric," Elsevier Journal of Systems and Software, Volume 98, 2014, pp. 25-43.

J107 Akram Hakiri, Pascal Berthoua, Aniruddha Gokhale, Douglas C. Schmidt, Gayraud Thierry, "Supporting SIP-based Data Distribution Service End-to-End QoS in WANs," the Elsevier Journal of Systems and Software, Volume 95, September 2014, pp. 100-121.

J106 Jules White, Douglas C. Schmidt, and Mani Golparvar-Fard, "Applications of Augmented Reality," IEEE Proceedings Special issue on Applications of Augmented Reality, Vol 102, No. 2., February 2014, pp. 120-123.

J105 Nickolas H. Guertin, Paul Bruhns, Douglas C. Schmidt, and Adam Porter, "Experiences Using Online War Games to Improve the Business of Naval Systems Acquisition," Cutter Journal of Information Technology Management, Vol. 27, No. 5, May 2014, pp 13-18.

J104 Michael McLendon, Bill Scherlis, and Douglas C. Schmidt, "Addressing Software Sustainment Challenges for the DoD," STSC CrossTalk, The Journal of Defense Software Engineering special issue on Legacy Systems Software, January, volume 27, number 1, 2014, pp. 27-32.

J103 Akram Hakiri, Pascal Berthoua, Aniruddha Gokhale, Douglas C. Schmidt, Gayraud Thierry, "Supporting End-to-end Scalability and Real-time Event Dissemination in the OMG Data Distribution Service over Wide Area Networks," Elsevier Journal of Systems and Software, volume 86, number 10, October, 2013, pp. 2574-2593.

J102 William Otte, Aniruddha Gokhale, and Douglas C. Schmidt, "Efficient and Deterministic Application Deployment in Component-based, Enterprise Distributed, Real-time, and Embedded

Systems," Elsevier Journal of Information and Software Technology, Vol. 55, No. 2, Feb 2013, 475-488.

J101 Dr. Douglas Schmidt, Anita Carleton, Erin Harper, Mary Ann Lapham, Ipek Ozkaya, and Linda Parker Gates, 'What Will It Take to Achieve Agility-at-Scale?", Cutter IT Journal, edited by Hillel Glazer, November 2012, pp. 34-39.

J100 Brian Dougherty, Jules White, and Douglas C. Schmidt, "Model-driven Auto-scaling of Green Cloud Computing Infrastructure," the Elsevier International Journal of Future Generation Computing Systems, Special Issue on Green Computing Systems, Volume 28, Number 2, February, 2012 Pages 371-378.

J99 Joe Hoffert, Douglas C. Schmidt, and Aniruddha Gokhale, "Evaluating Timeliness and Accuracy Trade-offs of Supervised Machine Learning for Adapting Enterprise DRE Systems in Dynamic Environments," the International Journal of Computational Intelligence Systems, Volume 4, Number 5, September-October 2011, pp. 806-816.

J98 James Hill, Pooja Varshneya, and Douglas C. Schmidt, "Evaluating Distributed Real-time and Embedded System Test Correctness using System Execution Traces," Central European Journal of Computer Science, Volume 1, Number 2, August 2011, pp. 167-184.

J97 Brian Dougherty, Jules White, and Douglas C. Schmidt, "Automated Software and Hardware Evolution Analysis for Distributed Real-time and Embedded Systems," the Central European Journal of Computer Science, Volume 1, Number 1, July 2011, pp. 36-57.

J96 James Hill, Hunt Sutherland, Paul Stodinger, Thomas Silveria, Douglas C. Schmidt, John Slaby, and Nikita Visnevski, "OASIS: An Architecture for Dynamic Instrumentation of Enterprise Distributed Real-time and Embedded Systems," the International Journal of Computer Systems Science and Engineering, Special Issue on Real-time Systems, Volume 26, Number 6, November 2011, pp. 413-430.

J95 Jules White, Brian Dougherty, Chris Thompson, Douglas C. Schmidt, "ScatterD: Spatial Deployment Optimization with Hybrid Heuristic/Evolutionary Algorithms," ACM Transactions on Autonomous and Adaptive Systems Special Issue on Spatial Computing, Volume 6 Issue 3, September 2011, 18:1-8:25.

J94 Jules White, Chris Thompson, Hamilton Turner, Brian Dougherty, and Douglas C. Schmidt, WreckWatch: Automatic Traffic Accident Detection and Notification with Smartphones, Journal of Mobile Networks and Applications, Volume 16 Issue 3, July 2011, Pages 285-303.

J93 Jules White, Brian Dougherty, Richard Schantz, Douglas C. Schmidt, Adam Porter, and Angelo Corsaro, "R&D Challenges and Solutions for Highly Complex Distributed Systems: a Middleware Perspective," the Springer Journal of Internet Services and Applications special issue on the Future of Middleware, Volume 2, Number 3, December 2011, pp. 1-8.

J92 Joe Hoffert, Aniruddha Gokhale, and Douglas C. Schmidt, "Autonomic Adaptation of Publish/Subscribe Middleware in Dynamic Environments," the International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS), 2(4), 1-24, October-December 2011, pp. 1-24.

J91 Joe Loyall, Matt Gillen, Aaron Paulos, Larry Bunch, Marco Carvalho, James Edmondson, Douglas C. Schmidt, Andrew Martignoni, and Asher Sinclair, "Dynamic Policy-Driven Quality of Service in Service-Oriented Information Management Systems," Wiley journal on Software: Practice and Experience, December 2011, volume 41, number 12, pp. 1459-1489.

J90 Michael Stal, Douglas C. Schmidt, and Will Otte, "Efficiently and Transparently Automating Scalable On-demand Activation and Deactivation of Services with the Activator Pattern," Software: Practice and Experience, special issue on Pattern Languages: Addressing Challenges, Edited by Mohamed Fayad and Shivanshu Singh, volume 41, number 10, October 2011, Wiley and Sons, pp. 1-16.

J89 Brian Dougherty, Jules White, Douglas C. Schmidt, Russell Kegley, and Jonathan Preston, "Deployment Optimization for Embedded Flight Avionics Systems," STSC CrossTalk, The Journal of Defense Software Engineering, November/December, volume 24, number 6, 2011, pp. 1-8.

J88 Brian Dougherty, Daniel Guymon, Douglas C. Schmidt, and Jules White, "Overcoming Cellular Connectivity Limitations with M2Blue Autonomic Distributed Data Caching," Autonomic Computing for Computer Society of India Magazine, CSI Communications, August 2011, pp. 12-15.

J87 Friedhelm Wolf, Jaiganesh Balasubramanian, Sumant Tambe, Aniruddha Gokhale, and Douglas C. Schmidt, Supporting Component-based Failover Units in Middleware for Distributed Real-time and Embedded Systems, the Elsevier Journal of System Architectures (JSA): Embedded Systems Design, Special Issue on Real-time and Embedded Systems, May, 2011 pp. 597-613.

J86 Jules White, David Benavides, Douglas C. Schmidt, Pablo Trinidad, Antonio Ruiz-Cortes, Brian Doughtery, "Automated Diagnosis of Feature Model Configurations," The Journal of Systems and Software, Special Issue on Software Product-lines, Volume 83, Issue 7, July, 2010, pp. 1094-1107.

J85 Jules White, Brian Doughtery, and Douglas C. Schmidt, "ASCENT: An Algorithmic Technique for Designing Hardware and Software in Tandem, IEEE Transactions on Software Engineering Special Issue on Search-based Software Engineering, November/December 2010 (vol. 36 no. 6), pp. 838-851.

J84 Joe Hoffert, Daniel Mack, and Douglas Schmidt, "Integrating Machine Learning Techniques to Adapt Protocols for QoS-enabled Distributed Real-time and Embedded Publish/Subscribe Middleware," International Journal of Network Protocols and Algorithms (NPA): Special Issue on Data Dissemination for Large-scale Complex Critical Infrastructures, Volume 2, Number 3, 2010, pp. 37-69.

J83 James Hill, James Edmondson, Aniruddha Gokhale, and Douglas C. Schmidt, "Tools for Continuously Evaluating Distributed System Qualities," IEEE Software, July/August, 2010, Volume 27, Number 4, pp. 65-71.

J82 James Edmondson and Douglas C. Schmidt, Multi-Agent Distributed Adaptive Resource Allocation (MADARA), International Journal of Communication Networks and Distributed Systems (IJCNDS), Special Issue on: Grid Computing, Edited by Michal Wozniak and Krzysztof Walkowiak, Volume 5, Number 3, 2010, pp. 229-245.

J81 Jules White, Christin Groba, Sibohan Clarke, Brian Dougherty, Chris Thompson, and Douglas C. Schmidt, "R&D Challenges and Solutions for Mobile Cyber-Physical Applications and Supporting Internet Services," the Springer Journal of Internet Services and Applications, Volume 1, Number 1, 2010, pp. 45-56.

J80 Jules White, Jeff Gray, and Douglas C. Schmidt, "Constraint-based Model Weaving," Transactions on Aspect-Oriented Software Development, Special Issue on Aspects and Model Driven Engineering, eds. Robert France and Jean-Marc Jezequel, pp. 153-190, Volume 5560, Number 6, 2009.

J79 Jules White, Harrison Strowd, and Douglas C. Schmidt, "Creating Self-healing Service Compositions with Feature Modeling and Microrebooting," the International Journal of Business Process Integration and Management (IJBPIM), Special issue on Model-Driven Service-Oriented Architectures, Inderscience Publishers, pp. 35-46, Volume 4, Number 1, 2009.

J78 Nishanth Shankaran, John Kinnebrew, Xenofon Koutsoukos, Chenyang Lu, Douglas C. Schmidt, and Gautam Biswas, "An Integrated Planning and Adaptive Resource Management Architecture for Distributed Real-time Embedded Systems," IEEE Transactions on Computers, Special Issue on Autonomic Network Computing, Special Issue on Autonomic Network Computing, volume 58, number 11, pp. 1485-1498, November 2009.

J77 Jules White, Brian Doughtery, and Douglas C. Schmidt, "Selecting Highly Optimal Architectural Feature Sets with Filtered Cartesian Flattening," the Journal of Software and Systems, Special Issue on Design Decisions and Design Rationale in Software Architecture, Volume 82, Issue 8, pp. 1268-1284, August 2009.

J76 Jules White, James, Hill, Sumant Tambe, Jeff Gray, Aniruddha Gokhale, and Douglas C. Schmidt "Improving Domain-specific Language Reuse through Software Product-line Configuration Techniques, IEEE Software Special Issue: Domain-Specific Languages and Modeling, vol. 26, no. 4, pp. 47-53, July/August 2009.

J75 Jules White and Douglas C. Schmidt, "Automating Deployment Planning with an Aspect Weaver," IET Software Journal Special Issue on Domain-specific Aspect Languages, Volume 3, Issue 3 , p. 167-183, June 2009.

J74 Shanshan Jiang, Yuan Xue, and Douglas C. Schmidt, "Minimum Disruption Service Composition and Recovery in Mobile Ad Hoc Networks, Elsevier Computer Networks Journal, Special Issue on Autonomic and Self-Organizing Systems, Volume 53, Issue 10, Pages 1649-1665, 2009.

J73 Nishanth Shankaran, Douglas C. Schmidt, Xenofon D. Koutsoukos, Yingming Chen, and Chenyang Lu, "Design and Performance Evaluation of an Adaptive Resource Management Framework for Distributed Real-time and Embedded Systems," EURASIP Journal on Embedded Systems (EURASIP JES): Special issue on Operating System Support for Embedded Real-Time Applications, Edited by Alfons Crespo, Ismael Ripoll, Michael Gonzalez Harbour, and Giuseppe Lipari, 2008, Pgs. 47-66.

J72 Aniruddha Gokhale, Krishnakumar Balasubramanian, Jaiganesh Balasubramanian, Arvind Krishna, and George T. Edwards, Gan Deng, Emre Turkay, Jeffrey Parsons, and Douglas C. Schmidt, Model Driven Middleware: A New Paradigm for Deploying and Provisioning Distributed Real-time and Embedded Applications, Elsevier Journal of Science of Computer Programming: Special Issue on Foundations and Applications of Model Driven Architecture (MDA), Edited by Mehmet Aksit, Volume 73, Issue 1, 1 September 2008, Pgs. 39-58.

J71 Nishanth Shankaran, Xenofon Koutsoukos, Chenyang Lu, Douglas C. Schmidt, and Yuan Xue, "Hierarchical Control of Multiple Resources in Distributed Real-time and Embedded Systems," the Springer Real-time Systems Journal, Volume 39, Numbers 1-3, August, 2008, pgs. 237-282.

J70 Douglas C. Schmidt, Angelo Corsaro, and Hans Van'T Hag, "Addressing the Challenges of Tactical Information Management in Net-Centric Systems with DDS," CrossTalk special issue on Distributed Software Development, pgs. 24-29, May 2008.

J69 Jules White, Douglas C. Schmidt, Egon Wuchner, and Andrey Nechypurenko, "Automatically Composing Reusable Software Components for Mobile Devices," Journal of the Brazilian Computer Society (JBCS), Special Issue in Software Reuse: Methods, Processes, Tools and Experiences, Sociedade Brasileira de Computacao, Porto Alegre, Volume 14, Number 1, pgs. 25-44, March, 2008.

J68 Jules White, Douglas C. Schmidt, Andrey Nechypurenko, and Egon Wuchner, "Model Intelligence: an Approach to Modeling Guidance," UPGRADE Journal, Volume 9, Number 2, pgs. 22-28, April 2008.

J67 Douglas C. Schmidt, "Beyond Objects: Evaluating Technologies for Developing Distributed Systems," Computer Society of India Communications, Special Issue on OO Technologies, edited by Debasish Jana, pgs. 30-37, February 2008.

J66 Jules White, Douglas C. Schmidt, and Aniruddha Gokhale, "Simplifying Autonomic Enterprise Java Bean Applications via Model-driven Engineering and Simulation," The Journal of Software and System Modeling, Volume 7, Number 1, February, 2008, pgs. 3-23.

J65 Cemal Yilmaz, Adam Porter, Arvind S. Krishna, Atif Memon, Douglas C. Schmidt, and Aniruddha Gokhale, "Reliable Effects Screening: A Distributed Continuous Quality Assurance Process for Monitoring Performance Degradation in Evolving Software Systems," IEEE Transactions on Software Engineering Vol. 33, No. 8, August 2007, pgs. 510-525.

J64 Frank Buschmann, Kevlin Henning, and Douglas C. Schmidt, "Past, Present, and Future Trends in Software Patterns," IEEE Software special issue on Patterns, Vol. 24, No. 4, July/August, 2007, pgs. 31-37.

J63 Patrick Lardieri, Jaiganesh Balasubramanian, Douglas C. Schmidt, Gautam Thaker, Aniruddha Gokhale, and Tom Damiano, A Multi-layered Resource Management Framework for Dynamic Resource Management in Enterprise DRE Systems, the Journal of Systems and Software: special issue on Dynamic Resource Management in Distributed Real-Time Systems, editors C. Cavanaugh and F. Drews and L. Welch, Vol 80, Issue 7, July 2007, pgs. 984-996.

J62 Janos Sztipanovits, John Bay, Larry Rohrbough, Shankar Sastry, Douglas C. Schmidt, Don Wilson, and Don Winters, "Escher: A New Technology Transitioning Model," IEEE Computer, Vol. 40, No. 3, March 2007, pgs. 90-92.

J61 Venkita Subramonian, Gan Deng, Christopher Gill, Jaiganesh Balasubramanian, Liang-Jui Shen, William Otte, Douglas C. Schmidt, Aniruddha Gokhale, and Nanbor Wang, "The Design and Performance of Component Middleware for QoS-enabled Deployment and Configuration of DRE Systems," Elsevier Journal of Systems and Software, Special Issue Component-Based Software Engineering of Trustworthy Embedded Systems, pp. 668-677, volume 80, number 5, March, 2007.

J60 Krishnakumar Balasubramanian, Jaiganesh Balasubramanian, Jeff Parsons, Aniruddha Gokhale, and Douglas C. Schmidt, "A Platform-Independent Component Modeling Language for

Distributed Real-time and Embedded Systems," Elsevier Journal of Computer and System Sciences, Volume 73, Number 2, March 2007, pgs. 171 - 185.

J59 Adam Porter, Atif Memon, Cemal Yilmaz, Douglas C. Schmidt, and Bala Natarajan, "Skoll: A Process and Infrastructure for Distributed Continuous Quality Assurance, IEEE Transactions on Software Engineering, 2007, Vol. 33, No. 2, February 2007, pgs. 124-141.

J58 Richard E. Schantz, Douglas C. Schmidt, Joseph P. Loyall, and Craig Rodrigues, "Controlling Quality-of-Service in Distributed Real-time and Embedded Systems via Adaptive Middleware," the Wiley Software Practice and Experience journal special issue on Experiences with Auto-adaptive and Reconfigurable Systems, co-editors Mehmet Aksit, Zied Choukair, and Tzilla Elrad, vol. 36, no. 11-12, September 2006, pgs. 1189 - 1208.

J57 Douglas C. Schmidt, "Model-Driven Engineering, IEEE Computer, Vol. 39. No. 2, February 2006, pp. 41-47.

J56 Arvind S. Krishna, Aniruddha Gokhale, Douglas C. Schmidt, John Hatcliff, and Venkatesh Prasad Ranganat, "Towards Highly Optimized Real-time Middleware for Software Product-line Architectures," ACM SIGBED Review, Volume 3, No. 1, January 2006, pgs. 12-16.

J55 Gabor Madl, Sherif Abdelwahed, and Douglas C. Schmidt, "Verifying Distributed Real-time Properties of Embedded Systems via Graph Transformations and Model Checking, Real-time Systems Journal, vol 33, no. 1-3, pgs. 77-100, July 2006.

J54 Cemal Yilmaz, Adam Porter, Atif Memon, Arvind S. Krishna, Douglas C. Schmidt, and Aniruddha Gokhale, Techniques and Processes for Improving the Quality and Performance of Open-Source Software, Software Process - Improvement and Practice Journal: Special Issue on Free/Open Source Software Processes, vol 11, no 2, May 2006, pgs. 163-176.

J53 Krishnakumar Balasubramanian, Arvind S. Krishna, Emre Turkay, Jaiganesh Balasubramanian, Jeff Parsons, Aniruddha Gokhale, and Douglas C. Schmidt, "Applying Model-Driven Development to Distributed Real-time and Embedded Avionics Systems, the International Journal of Embedded Systems, special issue on Design and Verification of Real-Time Embedded Software, April 2005.

J52 Arvind S. Krishna, Cemal Yilmaz, Adam Porter, Atif Memon, Douglas C. Schmidt, and Aniruddha Gokhale, "Distributed Continuous Quality Assurance Process for Evaluating QoS of Performance Intensive Software," Studia Informatica Universalis, Volume 4, No. 1, pp. 53-72, March 2005.

J51 Janos Sztipanovits, Gautam Biswas, Ken Frampton, Andy Gokhale, Larry Howard, Gabor Karsai, John Koo, Xenofon Koutsoukos, and Douglas C. Schmidt, "Introducing Embedded Software and Systems Education and Advanced Learning Technology in an Engineering Curriculum," ACM Transactions in Embedded Computing Systems - Special Issue on Education, edited by Alan Burns and Alberto Sangiovanni-Vincentelli, Vol 4, No. 1, pp. 549-568, August 2005.

J50 Arvind S. Krishna, Nanbor Wang, Balachandran Natarajan, Aniruddha Gokhale, Douglas C. Schmidt and Gautam Thaker, "CCMPerf: A Benchmarking Tool for CORBA Component Model Implementations", The International Journal of Time-Critical Computing Systems, Springer, Vol. 29, No. 2-3, pp. 281-308, March-April 2005.

J49 Chris Gill, Jeanna M. Gossett, David Corman, Joseph P. Loyall, Richard E. Schantz, Michael Atighetchi, and Douglas C. Schmidt, "Integrated Adaptive QoS Management in Middleware: An Empirical Case Study," The International Journal of Time-Critical Computing Systems, Springer, Vol. 29, Nos. 2-3, pp. 101-130, March-April 2005.

J48 Aniruddha Gokhale, Balachandran Natarajan, Douglas C. Schmidt, and Joseph Cross, "Towards Real-time Fault-Tolerant CORBA Middleware," Cluster Computing: the Journal on Networks, Software, and Applications Special Issue on Dependable Distributed Systems, edited by Alan George, Volume 7, Number 4, October 2004.

J47 Arvind S. Krishna, Cemal Yilmaz, Atif Memon, Adam Porter, Douglas C. Schmidt, Aniruddha Gokhale, and Balachandran Natarajan, "Preserving Distributed Systems Critical Properties: a Model-Driven Approach," the IEEE Software special issue on the Persistent Software Attributes, Nov/Dec 2004.

J46 Christopher Gill, Douglas C. Schmidt, Yamuna Krishnamurthy, Irfan Pyarali, Louis Mgeta, Yuanfang Zhang, and Stephen Torri, "Enhancing the Adaptivity of Distributed Real-time and Embedded Systems via Standard QoS-enabled Dynamic Scheduling Middleware," the Journal

of the Brazilian Computer Society (JCBS) special issue on Adaptive Software Systems, Volume 10, Number 1, pp. 19-30, 2004.

J45 Douglas C. Schmidt, Aniruddha Gokhale, and Balachandran Natarajan, "Frameworks: Why They Are Important and How to Apply Them Effectively," ACM Queue magazine, Volume 2, Number 5, July/August 2004.

J44 Douglas C. Schmidt, Richard Schantz, Aniruddha Gokhale, and Joe Loyall, "Middleware R&D Challenges for Distributed Real-time and Embedded Systems," ACM SIGBED Review, Volume 1, No. 1, April 2004.

J43 Angelo Corsaro and Douglas C. Schmidt, "The Design and Performance of Real-time Java Middleware," Special Issue on Middleware for the IEEE Transactions on Parallel and Distributed Systems, guest editor Rachid Guerraoui, Volume 14, Number 11, November 2003.

J42 Irfan Pyarali, Douglas C. Schmidt, and Ron Cytron, "Techniques for Enhancing Real-time CORBA Quality of Service," the IEEE Proceedings Special Issue on Real-time Systems, co-editors Yann-Hang Lee and C. M. Krishna, Volume 91, Number 7, July 2003.

J41 Nanbor Wang, Douglas C. Schmidt, Aniruddha Gokhale, Christopher D. Gill, Balachandran Natarajan, Craig Rodrigues, Joseph P. Loyall, and Richard E. Schantz, "Total Quality of Service Provisioning in Middleware and Applications," Elsevier Journal of Microprocessors and Microsystems, Volume 26, Number 9-10, January 2003.

J40 Christopher D. Gill, Douglas C. Schmidt, and Ron Cytron, "Multi-Paradigm Scheduling for Distributed Real-Time Embedded Computing," IEEE Proceedings Special Issue on Modeling and Design of Embedded Systems, Volume 91, Number 1, January, 2003.

J39 Aniruddha Gokhale, Bala Natarajan, Douglas C. Schmidt, and Nanbor Wang, "Modeling and Synthesis of Middleware Components," Communications of the ACM, special issue on Enterprise Components, Services and Business Rules, edited by Ali Arsanjani, October 2002.

J38 Douglas C. Schmidt and Carlos O'Ryan, "Patterns and Performance of Distributed Real-time and Embedded Publisher/Subscriber Architectures," the Journal of Systems and Software, Special Issue on Software Architecture – Engineering Quality Attributes, edited by Jan Bosch and Lars Lundberg, October 2002.

J37 Douglas C. Schmidt, "R&D Advances in Middleware for Distributed, Real-time and Embedded Systems," Communications of the ACM, Volume 45, Number 6, June 2002, edited by Gul Agha.

J36 Carlos O'Ryan, Douglas C. Schmidt, and J. Russell Noseworthy, "Patterns and Performance of a CORBA Event Service for Large-scale Distributed Interactive Simulations," International Journal of Computer Systems Science and Engineering, CRL Publishing, Volume 17, Number 2, March, 2002.

J35 Douglas C. Schmidt, Bala Natarajan, Aniruddha Gokhale, Nanbor Wang, and Chris Gill, "TAO: A Pattern-Oriented Object Request Broker for Distributed Real-time and Embedded Systems," IEEE Distributed Systems Online, Volume 3, Number 2, February, 2002.

J34 Douglas C. Schmidt, Rick Schantz, Mike Masters, Joseph Cross, David Sharp, and Lou DiPalma, "Towards Adaptive and Reflective Middleware for Network-Centric Combat Systems," CrossTalk, November, 2001.

J33 Nanbor Wang, Douglas C. Schmidt, Ossama Othman, and Kirthika Parameswaran, "Evaluating Meta-Programming Mechanisms for ORB Middleware," *IEEE Communications Magazine* special issue on "Evolving Communications Software: Techniques and Technologies," co-edited by Bill Opdyke and Algirdas Pakstas, Volume 39, Number 10, October, 2001.

J32 Nanbor Wang, Douglas C. Schmidt, Kirthika Parameswaran, and Michael Kircher, "Towards a Reflective Middleware Framework for QoS-enabled CORBA Component Model Applications," *IEEE Distributed Systems Online* special issue on Reflective Middleware, July, 2001.

J31 Chris Gill, David Levine, and Douglas C. Schmidt, "The Design and Performance of a Real-Time CORBA Scheduling Service," The *Real-time Systems, The International Journal of Time-Critical Computing Systems*, special issue on Real-Time Middleware, Kluwer Academic Publishers, guest editor Wei Zhao, Volume 20, Number 2, March 2001.

J30 Douglas C. Schmidt, Sumedh Mungee, Sergio Flores-Gaitan, and Aniruddha Gokhale, "Software Architectures for Reducing Priority Inversion and Non-determinism in Real-time Object Request Brokers," *Journal of Real-time Systems*, Kluwer, Vol. 21, No. 2, 2001.

J29 Ossama Othman, Carlos O'Ryan, and Douglas C. Schmidt, "An Efficient Adaptive Load Balancing Service for CORBA," *IEEE Distributed Systems Online*, March, 2001.

J28 Ossama Othman, Carlos O'Ryan, and Douglas C. Schmidt "The Design of an Adaptive CORBA Load Balancing Service, *IEEE Distributed Systems Online*, April, 2001.

J27 Carlos O'Ryan, Douglas C. Schmidt, Fred Kuhns, Marina Spivak, Jeff Parsons Irfan Pyarali, and David L. Levine, "Evaluating Policies and Mechanisms to Support Distributed Real-Time Applications with CORBA," *Concurrency and Computation: Practice and Experience* (Special Issue on Distributed Objects and Applications), Wiley and Sons, Vol. 13, No. 2, February, 2001.

J26 Douglas C. Schmidt, Vishal Kachroo, Yamuna Krisnamurthy, and Fred Kuhns, "Developing Next-generation Distributed Applications with QoS-enabled DPE Middleware," *IEEE Communications magazine*, edited by Abdi Modarressi and Sheshadri Mohan, Vol 17, No. 10, October, 2000.

J25 Douglas C. Schmidt and Fred Kuhns, "An Overview of the Real-time CORBA Specification," *IEEE Computer*, Special Issue on Object-Oriented Real-time Distributed Computing, edited by Eltefaat Shokri and Philip Sheu, June 2000.

J24 James Hu and Douglas C. Schmidt, Developing Flexible and High-performance Web Servers with Frameworks and Patterns, Symposium on Frameworks, *ACM Computing Surveys*, (Fayad and Wegner, eds.) Vol. 32(1es), March 2000.

J23 Fred Kuhns, Douglas C. Schmidt, Carlos O'Ryan, and David L. Levine, "Supporting High-performance I/O in QoS-enabled ORB Middleware," *Cluster Computing: the Journal on Networks, Software, and Applications*, Volume 3, Number 3, 2000.

J22 Irfan Pyarali, Carlos O'Ryan, Douglas C. Schmidt, Nanbor Wang, Vishal Kachroo, and Aniruddha Gokhale, "Using Principle Patterns to Optimize Real-time ORBs," *IEEE Concurrency*, Object-Oriented Systems track, edited by Murthy Devarakonda, Volumn 8, Number 1, January-March 2000.

J21 James Hu, Irfan Pyarali, and Douglas C. Schmidt, "The Object-Oriented Design and Performance of JAWS: A High-performance Web Server Optimized for High-speed Networks," The *Parallel and Distributed Computing Practices* journal, special issue on Distributed Object-Oriented Systems, edited by Maria Cobb, Vol. 3, No. 1, March 2000.

J20 Andy Gokhale and Douglas C. Schmidt, "Optimizing a CORBA IIOP Protocol Engine for Minimal Footprint Multimedia Systems," *IEEE Journal on Selected Areas in Communications* special issue on Service Enabling Platforms for Networked Multimedia Systems, September, 1999.

J19 Douglas C. Schmidt and Chris Cleeland, "Applying Patterns to Develop Extensible and Maintainable ORB Middleware," *IEEE Communications Magazine*, April, 1999.

J18 Irfan Pyarali and Douglas C. Schmidt, "An Overview of the CORBA Portable Object Adapter," Special Issue on CORBA in the ACM *StandardView* magazine, March, 1999.

J17 Prashant Jain, Seth Widoff, and Douglas C. Schmidt, "The Design and Performance of MedJava, A Distributed Electronic Medical Imaging System Developed with Java Applets and Web Tools," IEE/BCS *Distributed Systems Engineering Journal*, Volume 5, No. 4, December 1998.

J16 Douglas C. Schmidt, "Evaluating Architectures for Multi-threaded CORBA Object Request Brokers," *Communications of the ACM*, Special Issue on CORBA, ACM, edited by Krishnan Seetharaman, Volume 41, No. 10, October 1998.

J15 Andy Gokhale and Douglas C. Schmidt, "Measuring and Optimizing CORBA Latency and Scalability Over High-speed Networks," *IEEE Transactions on Computing*, April, 1998.

J14 Douglas C. Schmidt and James Hu, "Developing Flexible and High-performance Web Servers with Frameworks and Patterns," *Computing Surveys*, ACM, Vol. 29, March 1998.

J13 Douglas C. Schmidt, David Levine, and Sumedh Mungee, "The Design of the TAO Real-Time Object Request Broker," *Computer Communications*, Special Issue on Building Quality of Service into Distributed System, Elsevier Science, April, 1998.

J12 Guru Parulkar, Douglas C. Schmidt, Eileen Kraemer, Jon Turner, Anshul Kantawala, "An Architecture for Monitoring, Visualization, and Control and Gigabit Networks," *IEEE Network*, September/October, 1997.

J11 Douglas C. Schmidt, "Lessons Learned Building Reusable OO Frameworks for Distributed Software," *Communications of the ACM* Special Issue on OO Application Frameworks, ACM, Vol. 40, No. 10, October, 1997.

J10 Douglas C. Schmidt, "Applying Patterns to Meet the Challenges of Concurrent Software," *IEEE Concurrency*, Special Edition on Software Engineering for Parallel and Distributed Systems, Vol. 5, No. 3, August, 1997.

J9 Douglas C. Schmidt, Andy Gokhale, Tim Harrison, and Guru Parulkar, "A High-performance Endsystem Architecture for Real-time CORBA," *IEEE Communications Magazine*, Vol. 14, No. 2, February, 1997.

J8 Silvano Maffeis and Douglas C. Schmidt, "Constructing Reliable Distributed Communication Systems with CORBA," *IEEE Communications Magazine*, Vol. 14, No. 2, February, 1997.

J7 Douglas C. Schmidt, "Using Design Patterns to Develop Reuseable Object-Oriented Software," *ACM Computing Surveys*, Vol. 28, No. 4es, December 1996.

J6 Irfan Pyarali, Douglas C. Schmidt, and Tim Harrison, "Design and Performance of an Object-Oriented Framework for High-Speed Electronic Medical Imaging," *USENIX Computing Systems*, November/December, Vol. 9, No. 4, 1996.

J5 Douglas C. Schmidt, "A Family of Design Patterns for Application-level Gateways," *The Journal of Theory and Practice of Object Systems* (Special Issue on Patterns and Pattern Languages), Wiley and Sons, Vol. 2, No. 1, 1996.

J4 Douglas C. Schmidt, "Experience Using Design Patterns to Develop Reuseable Object-Oriented Communication Software," *Communications of the ACM Special Issue on Object-Oriented Experiences*, ACM, Vol. 38, No. 10, October, 1995, pp 65-74.

J3 Douglas C. Schmidt and Tatsuya Suda, "An Object-Oriented Framework for Dynamically Configuring Extensible Distributed Systems," *Distributed Systems Engineering Journal* (Special issue on Configurable Distributed Systems), IEE, Vol. 2, December, 1994, pp. 280-293.

J2 Douglas C. Schmidt, Donald F. Box, and Tatsuya Suda, "ADAPTIVE: A Dynamically Assembled Protocol Transformation, Integration, and eValuation Environment," *Journal of Concurrency: Practice and Experience*, Wiley and Sons, Ltd., Vol. 5, No. 4, June, 1993, pp. 269-286.

J1 Douglas C. Schmidt and Tatsuya Suda, "Transport System Architecture Services for High-Performance Communication Systems," *Journal of Selected Areas of Communications special-issue on Protocols for Gigabit Networks*, IEEE, Vol. 11, No. 4, May, 1993, pp. 489-506.

- **Book Publications and Book Chapters**

  – **Books Authored**

BA5 Frank Buschmann, Kevlin Henney, and Douglas C. Schmidt, *Pattern-Oriented Software Architecture: On Patterns and Pattern Languages*, Wiley and Sons, 2007.

BA4 Frank Buschmann, Kevlin Henney, and Douglas C. Schmidt, *Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing*, Wiley and Sons, 2007.

BA3 Steve Huston and Douglas C. Schmidt, *C++ Network Programming: Systematic Reuse with ACE and Frameworks*, Addison-Wesley Longman, 2003.

BA2 Douglas C. Schmidt and Steve Huston, *C++ Network Programming: Mastering Complexity with ACE and Patterns*, Addison-Wesley Longman, 2002.

BA1 Douglas C. Schmidt, Michael Stal, Hans Rohert, and Frank Buschmann, *Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects*, John Wiley and Sons, 2000.

  – **Book-length Reports Authored**

BR4 Werner Dahm, Douglas Schmidt, et al. "Cyber Situational Awareness," SAB-TR-12-01, 1 October 2012.

BR3 Alan Eckbreth, Charles Saff, Kevin Connolly, Natalie Crawford, Chris Eick, Mark Goorsky, Neil Kacena, David Miller, Robert Schafrik, Douglas Schmidt, "Sustaining Air Force Aging Aircraft into the 21st Century," SAB-TR-11-01 1 August 2011.

BR2 William Scherlis, Robert Behler, Barry Boehm, Lori Clarke, Michael Cusumano, Mary Ann Davidson, Larry Druffel, Russell Frew, James Larus, Greg Morrisett, Walker Royce, Douglas C. Schmidt, John Stenbit, Kevin Sullivan, *Critical Code Software Producibility for Defense*, Committee for Advancing Software-Intensive Systems Producibility, National Research Council of the National Academies, Washington, D.C.

BR1 Linda Northrop, Peter Feiler, Richard P. Gabriel, John Goodenough, Rick Linger, Tom Longstaff, Rick Kazman, Mark Klein, Linda Northrop, Douglas Schmidt, Kevin Sullivan, and Kurt Wallnau *Ultra-Large-Scale Systems: Software Challenge of the Future*, Software Engineering Institute, June 2006.

– **Books Edited**

BE3 Co-editor of *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, John Wiley & Sons, 1999 (co-editors are Mohamed Fayad and Ralph Johnson), ISBN 0-471-24875-4

BE2 Co-editor of *Implementing Application Frameworks: Object-Oriented Frameworks at Work*, John Wiley & Sons, 1999 (co-editors are Mohamed Fayad and Ralph Johnson), ISBN 0-471-25201-8

BE1 Co-editor of *Pattern Languages of Program Design*, Addison-Wesley, 1995 (co-editor is Jim Coplien, Bell Labs).

– **Book Chapters**

BC64 Michael Walker, Douglas C. Schmidt, and Abhishek Dubey, "Testing at Scale of IoT Blockchain Applications," *Role of Blockchain Technology in IoT Applications*, Volume 115, 1st Edition, edited by Shiho Kim, Ganesh Chandra Deka, and Peng Zhang, 2019.

BC63 Peng Zhang, Douglas C. Schmidt, Jules White, and Abhishek Dubey, "Consensus Mechanisms and Information Security Technologies," *Role of Blockchain Technology in IoT Applications*," Volume 115, 1st Edition, edited by Shiho Kim, Ganesh Chandra Deka, and Peng Zhang, 2019.

BC62 Peng Zhang, Douglas C. Schmidt, Jules White, and Gunther Lenz, "Blockchain Technology Use Cases in Healthcare," *Blockchain Technology: Platforms, Tools, and Use Cases*, edited by Ganesh Deka, 2018.

BC61 Michael Walker, Douglas C. Schmidt, and Jules White, "An Elastic Platform for Large-scale Assessment of Software Assignments for MOOCs (EPLASAM)," User-Centered Design Strategies for Massive Open Online Courses (MOOCs), edited by Ricardo Mendoza-Gonzalez, IGI Global, 2016.

BC60 Joseph Hoffert, Douglas C. Schmidt, and Aniruddha Gokhale, "Quantitative Productivity Analysis of a Domain-Specific Modeling Language," in the *Handbook of Research on Innovations in Systems and Software Engineering*, IGI Global, Aug 2014, pp. 313-344.

BC59 Will Otte, Douglas C. Schmidt, and Aniruddha Gokhale, "Performance and Scalability of a Large-scale Deployment and Configuration Framework," *The Performance of Open Source Applications*, edited by Tony Arkles and Tavish Armstrong, O'Reilly, 2013.

BC58 Chris Thompson, Jules White, and Douglas C. Schmidt, "Analyzing Mobile Application Software Power Consumption via Model-Driven Engineering," *Advances and Applications in Model-Driven Software Engineering*, edited by Vicente Garcia Diaz, IGI Global, 2013.

BC57 James Edmondson and Douglas C. Schmidt, "Towards Accurate Simulation of Large-Scale Systems via Time Dilation," *Real-time Simulation Technologies: Principles, Methodologies, and Applications*, edited by Katalin Popovici and Pieter J. Mosterman, CRC Press, 2012.

BC56 James Hill and Douglas C. Schmidt, "Using Test Clouds to Enable Continuous Integration Testing of Distributed Real-time and Embedded System Applications," *Software Testing in the Cloud: Perspectives on an Emerging Discipline.* Edited by Scott Tilley and Tauhida Parveen, IGI Global, 2012.

BC55 Angelo Corsaro and Douglas C. Schmidt, "The Data Distribution Service: The Communication Middleware Fabric for Scalable and Extensible Systems-of-Systems," *System of Systems*, edited by Adrian V. Gheorghe, InTech, 2012.

BC54 Joe Hoffert and Douglas C. Schmidt, "Improving Software Development Productivity for QoS Policy Configurations," *Model-Driven Domain Analysis and Software Development: Architectures and Functions*, a book edited by Janis Osis and Erika Asnina, 2011.

BC53 Nilabja Roy and Douglas C. Schmidt, "Model-Driven Performance Evaluation of Web Application Portals," *Model-Driven Domain Analysis and Software Development: Architectures and Functions*, a book edited by Janis Osis and Erika Asnina, 2011.

BC52 Brian Dougherty, Jules White, and Douglas C. Schmidt, "MDA-based Configuration of Distributed Real-time and Embedded Systems," *Model-Driven Domain Analysis and Soft-*

ware Development: Architectures and Functions, a book edited by Janis Osis and Erika Asnina, 2011.

BC51 Hamilton Turner, Jules White, Brian Dougherty, and Douglas C. Schmidt, "Building Mobile Sensor Networks Using Smartphones and Web Services: Ramifications and Development Challenges," *Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts*, edited by Maria Manuela Cruz-Cunha and Fernando Moreira, IGI Global, Hershey, PA, USA 2009.

BC50 Gan Deng, Jeff Gray, Douglas C. Schmidt, Yuehua Lin, Aniruddha Gokhale, and Gunther Lenz, "Evolution in Model-Driven Software Product-Line Architectures," *Software Applications: Concepts, Methodologies, Tools, and Applications*, edited by Pierre F. Tiako, 2009.

BC49 Jules White, Douglas C. Schmidt, Andrey Nechypurenko, and Egon Wuchner, "Reducing the Complexity of Modeling Large Software Systems," *Software Applications: Concepts, Methodologies, Tools, and Applications*, edited by Pierre F. Tiako, 2009.

BC48 Jules White, Brian Doughtery, Harrison Strowd, and Douglas C. Schmidt, "Using Filtered Cartesian Flattening and Microrebooting to Build Enterprise Applications with Self-adaptive Healing," *Software Engineering for Self-Adaptive Systems*, edited by Betty H. C. Cheng, Rogerio de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, 2009.

BC47 Jeff Gray, Sandeep Neema, Jing Zhang, Yuehua Lin, Ted Bapty, Aniruddha Gokhale, and Douglas C. Schmidt, "Concern Separation for Adaptive QoS Modeling in Distributed Real-Time Embedded Systems," *Behavioral Modeling for Embedded Systems and Technologies: Applications for Design and Implementation*, edited by Joa M. Fernandes and Luis Gomes, 2009.

BC46 Rick Schantz and Douglas C. Schmidt, "Middleware for Distributed Systems," Wiley *Encyclopedia of Computer Science and Engineering*, edited by Benjamin Wah, 2008.

B45 Jules White, Andrey Nechypurenko, Egon Wuchner, and Douglas C. Schmidt, "Reducing the Complexity of Optimizing Large-scale Systems by Integrating Constraint Solvers with Graphical Modeling Tools," *Designing Software-Intensive Systems: Methods and Principles*, Edited by Dr. Pierre F. Tiako, Langston University, OK, April 2008.

B44 Gan Deng, Douglas C. Schmidt, Aniruddha Gokhale, Jeff Gray, Yuehua Lin, and Gunther Lenz, "Evolution in Model-Driven Software Product-line Architecture, *Designing Software-Intensive Systems: Methods and Principles*, Edited by Dr. Pierre F. Tiako, Langston University, OK, April 2008.

B43 Daniel G. Waddington, Nilabja Roy, and Douglas C. Schmidt, "Dynamic Analysis and Profiling of Multi-threaded Systems," *Designing Software-Intensive Systems: Methods and Principles*, Edited by Dr. Pierre F. Tiako, Langston University, OK, April 2008.

B42 Krishnakumar Balasubramanian, Douglas C. Schmidt, Zoltan Molnar, and Akos Ledeczi, "System Integration via Model-Driven Engineering," *Designing Software-Intensive Systems: Methods and Principles* Edited by Dr. Pierre F. Tiako, Langston University, OK, April 2008.

B41 James Hill, Douglas C. Schmidt,, and John Slaby, "System Execution Modeling Tools for Evaluating the Quality of Service of Enterprise Distributed Real-time and Embedded System",*Designing Software-Intensive Systems: Methods and Principles*, Edited by Dr. Pierre F. Tiako, Langston University, OK, April 2008.

B40 Gan Deng, Chris Gill, Douglas C. Schmidt, and Nanbor Wang, "QoS-enabled Component Middleware for Distributed Real-Time and Embedded Systems," Handbook of Real-Time and Embedded Systems (I. Lee, J. Leung, and S. Son, eds.), CRC Press, 2007.

B39 William Otte and Douglas C. Schmidt, "Labor-Saving Architecture: an Object-Oriented Framework for Networked Software," *Beautiful Code*, edited by Greg Wilson, O'Reilly, 2007.

BC38 Irfan Pyarali, Carlos O'Ryan, and Douglas C. Schmidt, "A Pattern Language for Efficient, Predictable, Scalable, and Flexible Dispatching Components," Pattern Language of Program Design 5 book, Addison-Wesley, Reading, MA, 2006.

BC37 Douglas C. Schmidt, Krishnakumar Balasubramanian, Arvind S. Krishna, Emre Turkay, and Aniruddha Gokhale, Model-driven Development of Component-based Distributed Real-time and Embedded Systems, Model Driven Engineering for Distributed Real-time and Embedded Systems, edited by Sebastien Gerard, Joel Champea, and Jean-Philippe Babau, Hermes, 2005.

BC36 Gabriele A. Trombetti, Aniruddha Gokhale, Douglas C. Schmidt, John Hatcliff, Gurdip Singh, and Jesse Greenwald, "An Integrated Model-driven Development Environment for Composing and Validating Distributed Real-time and Embedded Systems," *Model-driven Software Development: Volume II of Research and Practice in Software Engineering*, edited by Sami Beydeda, Matthias Book, and Volker Gruhn, Springer-Verlag, 2005.

BC35 Arvind S. Krishna, Douglas C. Schmidt, Ray Klefstad, and Angelo Corsaro, "Real-time CORBA Middleware," in *Middleware for Communications*, edited by Qusay Mahmoud, Wiley and Sons, New York, 2003.

BC34 Nanbor Wang, Douglas C. Schmidt, Aniruddha Gokhale, Craig Rodrigues, Balachandran Natarajan, Joseph P. Loyall, Richard E. Schantz, and Christopher D. Gill, "QoS-enabled Middleware," in *Middleware for Communications*, edited by Qusay Mahmoud, Wiley and Sons, New York, 2003.

BC33 Aniruddha Gokhale, Douglas C. Schmidt, Balachandran Natarajan, Jeff Gray, and Nanbor Wang, "Model Driven Middleware," in *Middleware for Communications*, edited by Qusay Mahmoud, Wiley and Sons, New York, 2003.

BC32 Jeff Gray, Janos Sztipanovits, Ted Bapty, Sandeep Neema, Aniruddha Gokhale, and Douglas C. Schmidt, "Two-level Aspect Weaving to Support Evolution of Model-Based Software," in *Aspect-Oriented Software Development,* edited by Robert Filman, Tzilla Elrad, Mehmet Aksit, and Siobhan Clarke, Reading, Massachusetts: Addison-Wesley, 2003.

BC31 Joseph K. Cross and Douglas C. Schmidt "Applying the Quality Connector Pattern to Optimize Distributed Real-time and Embedded Middleware," *Patterns and Skeletons for Parallel and Distributed Computing*, edited by Fethi Rabhi and Sergei Gorlatch, Springer Verlag, 2002.

BC30 Richard E. Schantz and Douglas C. Schmidt, "Middleware for Distributed Systems: Evolving the Common Structure for Network-centric Applications," *Encyclopedia of Software Engineering*, edited by John Marciniak and George Telecki, Wiley and Sons, 2001.

BC29 Sumedh Mungee, Nagarajan Surendran, Yamuna Krishnamurthy, and Douglas C. Schmidt "The Design and Performance of a CORBA Audio/Video Streaming Service," *Multimedia Networking: Technology, Management, and Applications*, edited by Mahbubur Syed, Idea Group Publishing, Hershey, USA, 2001.

BC28 Nanbor Wang, Douglas C. Schmidt, and Carlos O'Ryan "An Overview of the CORBA Component Model," *Component-Based Software Engineering*, (George Heineman and Bill Councill, eds.), Addison-Wesley, Reading, MA, 2001.

BC27 Douglas C. Schmidt, "Applying a Pattern Language to Develop Application-Level Gateways," Design Patterns in Communications, (Linda Rising, ed.), Cambridge University Press, 2000.

BC26 Douglas C. Schmidt and Chris Cleeland, "Applying a Pattern Language to Develop Extensible ORB Middleware," Design Patterns in Communications, (Linda Rising, ed.), Cambridge University Press, 2000.

BC25 Carlos O'Ryan, Fred Kuhns, Douglas C. Schmidt, and Jeff Parsons, "Applying Patterns to Develop a Pluggable Protocols Framework for Object Request Broker Middleware," Design Patterns in Communications, (Linda Rising, ed.), Cambridge University Press, 2000.

BC24 David L. Levine, Christopher D. Gill, and Douglas C. Schmidt, "Object Lifecycle Manager – A Complementary Pattern for Controlling Object Creation and Destruction," Design Patterns in Communications, (Linda Rising, ed.), Cambridge University Press, 2000.

BC23 Douglas C. Schmidt, "A Family of Design Patterns For Flexibly Configuring Network Services in Distributed Systems," Design Patterns in Communications, (Linda Rising, ed.), Cambridge University Press, 2000.

BC22 James Hu and Douglas C. Schmidt, "JAWS: A Framework for High-performance Web Servers," *Object-Oriented Application Frameworks* book, John Wiley & Sons, October, 1999.

BC21 Chris Cleeland and Douglas C. Schmidt, "External Polymorphism, An Object Structural Pattern for Transparently Extending C++ Concrete Data Types" in C++ Gems II, (Robert Martin, ed.), SIGS, NY, 1999.

BC20 Douglas C. Schmidt, "GPERF: A Perfect Hash Function Generator" in C++ Gems II, (Robert Martin, ed.), SIGS, NY, 1999.

BC19 Douglas C. Schmidt, Tim H. Harrison, and Nat Pryce, "Thread-specific Storage: an

Object Behavioral Pattern for Efficiently Accessing per-Thread State" in C++ Gems II, (Robert Martin, ed.), SIGS, NY, 1999.

BC18 Irfan Pyarali, Tim Harrison, Douglas C. Schmidt, and Thomas Jordan, "Proactor: an Object Behavioral Pattern for Demultiplexing and Dispatching Handlers for Asynchronous Events," *Pattern Languages of Program Design,* (Harrison, Foote, and Rohnert, eds.), Addison-Wesley, Reading, MA, 1999.

BC17 Douglas C. Schmidt and Paul Stephenson, "Using Design Patterns to Evolve System Software from UNIX to Windows NT," In *The Patterns Handbook*, (Linda Rising, ed.), Cambridge University Press, 1998.

BC16 Douglas C. Schmidt, David L. Levine, and Chris Cleeland, "Architectures and Patterns for High-performance, Real-time CORBA Object Request Brokers," *Advances in Computers*, Academic Press, Ed., Marvin Zelkowitz, Volume 48, July 1999.

BC15 Douglas C. Schmidt and Tatsuya Suda, "A Framework for Measuring the Performance of Alternative Process Architectures for Parallel Communication Subsystems," in *Network Performance Modeling and Simulation*, Walrand, Bagchi, and Zobrist, Ed., Gordon and Breach Publishers, 1998.

BC14 Douglas C. Schmidt, "Applying Design Patterns and Frameworks to Develop Object-Oriented Communication Software," The Handbook of Programming Languages, Volume I, edited by Peter Salus, MacMillan Computer Publishing, 1997.

BC13 Chris Cleeland, Douglas C. Schmidt, and Tim H. Harrison, "External Polymorphism – An Object Structural Pattern for Transparently Extending Concrete Data Types," *Pattern Languages of Program Design,* (Martin, Buschmann, and Riehl, eds.), Addison-Wesley, Reading, MA, 1997.

BC12 Timothy H. Harrison, Douglas C. Schmidt, and Irfan Pyarali, "Asynchronous Completion Token – An Object Behavioral Pattern for Efficient Asynchronous Event Handling," *Pattern Languages of Program Design,* (Martin, Buschmann, and Riehl, eds.), Addison-Wesley, Reading, MA, 1997.

BC11 Douglas C. Schmidt and Timothy H. Harrison, "Double-Checked Locking – A Optimization Pattern for Efficiently Initializing and Accessing Thread-safe Objects," *Pattern Languages of Program Design,* (Martin, Buschmann, and Riehl, eds.), Addison-Wesley, Reading, MA, 1997.

BC10 Douglas C. Schmidt, "Acceptor and Connector – A Family of Object Creational Patterns for Initializing Communication Services," *Pattern Languages of Program Design,* (Martin, Buschmann, and Riehl, eds.), Addison-Wesley, Reading, MA, 1997.

BC9 Douglas C. Schmidt and Paul Stephenson, "Using Design Patterns to Evolve System Software from UNIX to Windows NT," In *Wisdom of the Gurus*, (Charles Bowman, ed.), Cambridge University Press, 1996.

BC8 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Distributed Programming Techniques" in *Wisdom of the Gurus*, (Charles Bowman, ed.), Cambridge University Press, 1996.

BC7 Douglas C. Schmidt, "A Case Study in C++ Design Evolution" in *C++ Gems,* (Stanley Lippman, ed.), SIGS, NY, 1996, pp. 99-120.

BC6 Douglas C. Schmidt and Steve Vinoski, "Distributed Object Computing in C++" in *C++ Gems,* (Stanley Lippman, ed.), SIGS, NY, 1996, pp. 303-316.

BC5 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Distributed Programming Techniques" in *C++ Gems,* (Stanley Lippman, ed.), SIGS, NY, 1996, pp. 316-336.

BC4 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Server Programming Techniques" in *C++ Gems,* (Stanley Lippman, ed.), SIGS, NY, 1996, pp. 337-362.

BC3 Douglas C. Schmidt and Charles D. Cranor, "Half-Sync/Half-Async: A Architectural Pattern for Efficient and Well-structured Concurrent I/O" in *Pattern Languages of Program Design,* (Coplien, Vlissides, and Kerth, eds.), Addison-Wesley, Reading, MA, 1996.

BC2 R. Greg Lavender and Douglas C. Schmidt, "Active Object: An Object Behavioral Pattern for Concurrent Programming," in *Pattern Languages of Program Design,* (Coplien, Vlissides, and Kerth, eds.), Addison-Wesley, Reading, MA, 1996.

BC1 Douglas C. Schmidt, "Reactor: An Object Behavioral Pattern for Concurrent Event Demultiplexing and Event Handler Dispatching," *Pattern Languages of Program Design,* (Addison-Wesley, 1995), edited by James O. Coplien and Douglas C. Schmidt.

- **Refereed Conference Publications**

C199 Quchen Fu, Zhongwei Teng, Jules White, Maria E. Powell, and Douglas C. Schmidt, "Fastaudio: A Learnable Audio Front-end for Spoof Speech Detection," proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 22-27th, Singapore.

C198 Quchen Fu, Zhongwei Teng, Jules White and Douglas Schmidt, "Translating Natural Language to Bash Code," proceedings of the IEEE 2021 International Conference on Machine Learning and Applications, December 13-16, 2021.

C197 Zhongwei Teng, Quchen Fu, Jules White, and Douglas C. Schmidt, "Analyzing the Feasibility of Generating Data Visualizations from Hand-drawn Sketches Using Deep Learning," proceedings of the 20th IEEE International Conference on Machine Learning and Applications, December 13-16, 2021.

C196 Gabriela Gresenz, Jules White, and Douglas C. Schmidt, "An Off-Road Terrain Dataset Including Images Labeled With Measures of Terrain Roughness," proceedings of the IEEE International Conference on Autonomous Systems (IEEE ICAS 2021), Montreal, Canada, August 11-13, 2021.

C195 Peng Zhang, Douglas C. Schmidt, and Jules White, "A Pattern Sequence for Designing Blockchain-Based Healthcare Information Technology Systems," the 26th Pattern Languages of Programming conference, October 7-10, 2019, Ottawa, Ontario, Canada.

C194 Nick Guertin, Douglas C. Schmidt, and Harry Levinson, "All Capability to All Boats, All the Time: Adding Options to the TI/APB Process," Proceedings of the 2018 Joint Undersea Warfare Technology Fall Conference, Groton, CT, September 18th, 2018.

C193 Zhongwei Teng, Peng Zhang, Xiao Li, William Nock, Marcelino Rodriguez-Cancio, Denis Gilmore, Jules White, Douglas C. Schmidt, and Jonathan C. Nesbitt, "Authentication and Usability in mHealth Apps," proceedings of the 2018 IEEE International Conference on E-health Networking, Application and Services (Healthcom), 17-20 September 2018, Ostrava, Czech Republic (winner of the "Outstanding Paper" award).

C192 Peng Zhang, Douglas C. Schmidt, Jules White, and Shelagh A. Mulvaney, Towards Precision Behavioral Medicine with IoT: Iterative Design and Optimization of a Self-Management Tool for Type 1 Diabetes," proceedings of the 2018 IEEE International Conference on Healthcare Informatics (ICHI 2018), New York, NY, USA, June 4-7, 2018.

C191 Nick Guertin, Douglas C. Schmidt, and Bill Scherlis, "Capability Composition and Data Interoperability to Achieve More Effective Results than DoD System-of-Systems Strategies," proceedings of 15th Annual Acquisition Research Symposium, May 9-10, 2018 in Monterey, CA.

C190 Peng Zhang, Douglas C. Schmidt, Jules White, and Gunther Lenz, "Metrics for Assessing Blockchain-based Healthcare Decentralized Apps," Proceedings of the IEEE Healthcom 2017, October 12-15, 2017, Dalian, China.

C189 Peng Zhang, Jules White, Douglas C. Schmidt, and Gunther Lenz, "Design of Blockchain-Based Apps Using Familiar Software Patterns to Address Interoperability Challenges in Healthcare," the 24th Pattern Languages of Programming (PLoP) conference, October 22-25, 2017, Vancouver, Canada.

C188 Aron Laszka, Michael Walker, Abhishek Dubey and Douglas Schmidt, "Providing Privacy, Safety, and Security in IoT-Based Transactive Energy Systems using Distributed Ledgers," The 7th International Conference on the Internet of Things (IoT 2017), October 22-25, 2017, Linz, Austria.

C187 Fangzhou Sun, Peng Zhang, Jules White, Douglas C. Schmidt, Jacob Staples, and Lee Krause, A Feasibility Study of Autonomically Detecting In-process Cyber-Attacks, Proceedings of the 3rd IEEE International Conference on Cybernetics (CYBCONF-2017), Special Session on Cyber Security.

C186 Peng Zhang, Jules White, Douglas C. Schmidt, and Tom Dennis, "Discussions of a Preliminary Hand Hygiene Compliance Monitoring Application-as-a-Service", 10th International Conference on Health Informatics - HEALTHINF 2017, 21 - 23 February, 2017, Porto, Portugal.

C185 Peng Zhang, Jules White, Douglas C. Schmidt, and Tom Dennis, "Applying Machine Learning Methods to Predict Hand Hygiene Compliance Characteristics," Proceedings of the Biomedical and Health Informatics Conference, Orlando, Florida, February 16-19, 2017.

C184 Subhav Pradhan, Shweta Khare, Fangzhou Sun, Abhishek Dubey, Janos Sallai, Aniruddha Gokhale, Douglas Schmidt, Martin Lehofer, and Monika Sturm, "Towards a Distributed and Resilient Platform for Smart City Systems," First IEEE/ACM Symposium on Edge Computing, October 27-28, 2016, Washington DC.

C183 Peng Zhang, Jules White, and Douglas C. Schmidt, "HoliCoW: Automatically Breaking Team-based Software Projects to Motivate Student Testing," Proceedings to the Software Engineering Education and Training track at the 38th International Conference on Software Engineering Austin, TX, May 14 - 22, 2016.

C182 Nickolas H. Guertin, Robert Sweeney, and Douglas C. Schmidt, "How the Navy Can Use Open Systems Architecture to Revolutionize Capability Acquisition," Naval Postgraduate School's Acquisition Research Symposium, May 13th 2015, Monterey, CA.

C181 Nickolas H. Guertin, Robert Sweeney, and Douglas C. Schmidt, "Benefits of Applying an Open Systems Architecture Approach," 17th Systems Engineering Conference, National Defense Industry Association, October 27-30, 2014, Washington DC.

C180 Kyoungho An, Aniruddha Gokhale, Sumant Tambe, Gerardo Pardo-Castellote, and Douglas C. Schmidt, "Content-based Filtering Discovery Protocol (CFDP): Scalable and Efficient OMG DDS Discovery Protocol," 8th ACM International Conference on Distributed Event-Based Systems, Mumbai, India, May 26-29, 2014.

C179 Balakrishnan Dasarathy, Kevin Sullivan, Douglas C. Schmidt, Douglas H. Fisher, and Adam Porter, "The Past, Present, and Future of MOOCs and Their Relevance to Software Engineering," 36th ACM/IEEE International Conference on Software Engineering, Hyderabad, India, May 31 - June 7th, 2014.

C178 Rick Leathart, Adam Porter, Douglas Schmidt, Michael O'Hare, Harry Crisp, and Barry Laird, "Capability-Based Technical Reference Frameworks for Open System Architecture Implementations," Systems Engineering Conference (SEDC) 20134, Washington DC, April 3rd and 4th, 2014, Chantilly, VA.

C177 Douglas C. Schmidt, Chris Gill, and Jules White, "Elastic Infrastructure to Support Computing Clouds for Large-scale Cyber-Physical Systems," Proceedings of the International Symposium on Object-Oriented Real-time Distributed Computing (ISORC), June 2014, Reno, Nevada.

C176 Douglas C. Schmidt and Zach McCormick, "Creating and Teaching a MOOC on Pattern-Oriented Software Architecture for Concurrent and Networked Software," Proceedings of the WaveFront Forum at the SPLASH 2013 conference, October 2013, Indianapolis, IN.

C175 Zach McCormick and Douglas C. Schmidt, "Data Synchronization Patterns in Mobile Application Design," Proceedings of the Pattern Languages of Programs (PLoP) 2012 conference, October 19-21, 2012, Tucson, Arizona.

C174 James Edmondson, Aniruddha Gokhale and Douglas Schmidt, "Approximation Techniques for Maintaining Real-time Deployments Informed by User-provided Dataflows Within a Cloud," 31st International Symposium on Reliable Distributed Systems (SRDS 2012), 8th-11th October 2012, Irvine, California.

C173 James Edmondson, Douglas C. Schmidt, and Aniruddha Gokhale "QoS-enabled Distributed Mutual Exclusion in Public Clouds," Proceedings of the 1st International Symposium on Secure Virtual Infrastructures (DOA-SVI'11), October 17-19, 2011, Crete, Greece.

C172 Brian Dougherty, Jules White, Russell Kegley, Jonathan Preston, Douglas C. Schmidt, and Aniruddha Gokhale, "Optimizing Integrated Application Performance with Cache-aware Metascheduling," Proceedings of the 1st International Symposium on Secure Virtual Infrastructures (DOA-SVI'11), October 17-19, 2011, Crete, Greece.

C171 Akram Hakiri, Aniruddha Gokhale, Douglas C. Schmidt, Berthou Pascal, Joe Hoffert, and Gayraud Thierry, "A SIP-based Network QoS Provisioning Framework for Cloud-hosted DDS Applications," Proceedings of the 1st International Symposium on Secure Virtual Infrastructures (DOA-SVI'11), October 17-19, 2011, Crete, Greece.

C170 James Hill and Douglas C. Schmidt, "Experiences with Service-Oriented Middleware for Dynamic Instrumentation of Enterprise Distributed Real-time and Embedded Systems," Proceedings of the 1st International Symposium on Secure Virtual Infrastructures (DOA-SVI'11), October 17-19, 2011, Crete, Greece.

C169 Will Otte, Aniruddha Gokhale, Douglas C. Schmidt, and Johnny Willemsen, "Infrastructure for Component-based DDS Application Development," proceedings of the Tenth International Conference on Generative Programming and Component Engineering (GPCE'11), October 22-23, 2011 Portland, Oregon, USA.

C168 Will Otte, Aniruddha Gokhale, and Douglas C. Schmidt, Techniques for Predictable Deployment Latencies in Large-scale Component-based Distributed Real-time and Embedded Systems, Proceedings of the 14th International ACM SIGSOFT Symposium on Component Based Software Engineering (CBSE-2011), June 21th - 23th, 2011, in Boulder, Colorado, USA.

C167 Chris Thompson, Hamilton Turner, Jules White, and Douglas C. Schmidt, Analyzing Mobile Application Software Power Consumption via Model-Driven Engineering, Proceedings of the 1st International Conference on Pervasive and Embedded Computing and Communication Systems, Algarve, Portugal, March 5-7, 2011.

C166 Joe Hoffert, Douglas C. Schmidt, Aniruddha Gokhale, "Adapting Distributed Real-time and Embedded Publish/Subscribe Middleware for Cloud-Computing Environments," Proceedings of the ACM/IFIP/USENIX 11th International Middleware Conference, Bangalore, India, November 30-Dec 3, 2010.

C165 Joe Hoffert and Douglas Schmidt, "Evaluating Supervised Machine Learning for Adapting Enterprise DRE Systems," Proceedings of the International Symposium on Intelligence Information Processing and Trusted Computing (IPTC 2010), Huanggang City, China, October 28-29, 2010.

C164 J. Benjamin Gotow, Krzysztof Zienkiewicz, Jules White, and Douglas C. Schmidt, "Addressing Challenges in Delivering Augmented Reality Applications to Smartphones," Proceedings of the Third International ICST Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2010), June 30-July 2, 2010, Chicago, IL.

C163 Chris Thompson, Jules White, Brian Dougherty, Adam Albright, and Douglas C. Schmidt, "Using Smartphones and Wireless Mobile Networks to Detect Car Accidents and Provide Situational Awareness to Emergency Responders," Proceedings of the Third International ICST Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2010), June 30-July 2, 2010, Chicago, IL.

C162 James H. Hill, Hunt Sutherland, Douglas C. Schmidt, Thomas Silveria, John M. Slaby, Paul Staudinger, and Nikita A. Visnevski, "OASIS: A Service-Oriented Architecture for Dynamic Instrumentation of Enterprise Distributed Real-time and Embedded Systems," Proceedings of the 13th International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC '10), May 5-6, 2010, Carmona, Spain.

C161 Joseph P. Loyall, Matthew Gillen, Aaron Paulos, Larry Bunch, Marco Carvalho, James Edmondson, Pooja Varshneya, Douglas C. Schmidt, Andrew Martignoni, "Dynamic Policy-Driven Quality of Service in Service-Oriented Systems," Proceedings of the 13th International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC '10), May 5-6, 2010, Carmona, Spain.

C160 Jaiganesh Balasubramanian, Aniruddha Gokhale, Abhishek Dubey, Friedhelm Wolf, Chenyang Lu, Chris Gill, and Douglas C. Schmidt, "Middleware for Resource-Aware Deployment and Configuration of Fault-tolerant Real-time Systems," Proceedings of the 16th IEEE Real-Time and Embedded Technology and Applications Symposium Stockholm, Sweden, April 12 - 15, 2010.

C159 John S. Kinnebrew, Daniel L. C. Mack, Gautam Biswas, Douglas C. Schmidt, "Coordination of Planning and Scheduling Techniques for a Distributed, Multi-level, Multi-agent System", Proceedings of the 2nd International Conference on Agents and Artificial Intelligence (ICAART), Valencia, Spain, January 22-24, 2010.

C158 Nilabja Roy, Yuan Xue, Aniruddha Gokhale, Larry Dowdy and Douglas C. Schmidt, "A Component Assignment Framework for Improved Capacity and Assured Performance inWeb Portals," Proceedings of the 11th International Symposium on Distributed Objects, Middleware, and Applications (DOA'09) Vilamoura, Algarve-Portugal, Nov 01 - 03, 2009.

C157 Joe Hoffert, Douglas C. Schmidt, and Aniruddha Gokhale, "Evaluating Transport Protocols for Real-time Event Stream Processing Middleware and Applications," Proceedings of the 11th International Symposium on Distributed Objects, Middleware, and Applications (DOA'09) Vilamoura, Algarve-Portugal, Nov 01 - 03, 2009.

C156 Joe Hoffert and Douglas C. Schmidt, "Maintaining QoS for Publish/Subscribe Middleware in Dynamic Environments," Fast Abstract, 3rd ACM International Conference on Distributed Event-Based Systems (DEBS 2009) July 6-9, 2009 - Nashville, TN, USA.

C155 Nilabja Roy, Larry Dowdy, and Douglas C. Schmidt, "The Impact of Variability on Soft Real-Time System Scheduling," Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2009), Beijing, China, August 24-26, 2009.

C154 Friedhelm Wolf, Jaiganesh Balasubramanian, Aniruddha Gokhale, and Douglas C. Schmidt, "Component Replication based on Failover Units," Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2009), Beijing, China, August 24-26, 2009.

C153 Jules White, David Benavides, Brian Dougherty, Douglas C. Schmidt, "Automated Reasoning for Multi-step Software Product-line Configuration Problems," Proceedings of the 13th International Software Product Line Conference (SPLC 2009) August 24-28, 2009, San Francisco, CA.

C152 Joseph Loyall, Marco Carvalho, Douglas Schmidt, Matthew Gillen, Andrew Martignoni III, Larry Bunch, James Edmondson, and David Corman, "QoS Enabled Dissemination of Managed Information Objects in a Publish-Subscribe-Query Information Broker," the SPIE Defense Transformation and Net-Centric Systems conference, April, 2009, Orlando, FL.

C151 Brian Dougherty, Jules White, Jaiganesh Balasubramanian, Chris Thompson, and Douglas C. Schmidt, "Deployment Automation with BLITZ," Proceedings of the Emerging Results track at the 31st International Conference on Software Engineering, Vancouver, Canada, May 16-24, 2009.

C150 Brian Dougherty, Jules White, Chris Thompson, and Douglas C. Schmidt, "Automating Hardware and Software Evolution Analysis," Proceedings of the 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS), April 13-16, 2009, San Francisco, CA USA.

C149 James H. Hill, Hamilton A. Turner, James R. Edmondson, and Douglas C. Schmidt, "Unit Testing Non-functional Concerns of Component-based Distributed Systems," Proceedings of the 2nd International Conference on Software Testing, Verification, and Validation, April 1 - 4, 2009, Denver, Colorado.

C148 John S. Kinnebrew, William R. Otte, Nishanth Shankaran, Gautam Biswas, and Douglas C. Schmidt, "Intelligent Resource Management and Dynamic Adaptation in a Distributed Real-time and Embedded Sensor Web System," Proceedings of the 12th International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC '09), Tokyo, Japan, March 17-20, 2009.

C147 Jaiganesh Balasubramanian, Sumant Tambe, Chenyang Lu, Aniruddha Gokhale, Christopher Gill, and Douglas C. Schmidt, "Adaptive Failover for Real-time Middleware with Passive Replication, Proceedings of the 15th Real-time and Embedded Applications Symposium (RTAS) 2009, San Francisco, CA, United States, April 13 - 16, 2009.

C146 William R. Otte, John. S. Kinnebrew, Douglas C. Schmidt, and Gautam Biswas, "A Flexible Infrastructure for Distributed Deployment in Adaptive Sensor Webs," Proceedings of the 2009 IEEE Aerospace Conference, Big Sky, Montana, March 2009.

C145 Nanbor Wang, Douglas C. Schmidt, Angelo Corsaro, and Hans Van'T Hag, "Toward an Adaptive Data Distribution Service for Dynamic Large-Scale Network-Centric Operation and Warfare Systems," Proceedings of the 2008 Military Communications Conference, November 17-19, 2008 in San Diego, CA.

C144 Joe Hoffert, Douglas C. Schmidt, and Aniruddha Gokhale, "DQML: A Modeling Language for Configuring Publish/Subscribe Quality of Service Policies," Proceedings of the Distributed Objects, Middleware, and Applications (DOA'08), Monterrey, Mexico, Nov 10 - 12, 2008.

C143 Nilabja Roy, Akshay Dabholkar, Nathan Hamm, Larry Dowdy and Douglas Schmidt, "Modeling Software Contention using Colored Petri Nets," Proceedings of the 16th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), September 8-10 2008, Baltimore, MD.

C142 Jules White, Douglas C. Schmidt, David Benavides, Pablo Trinidad, Antonio Ruiz-Cortez, "Automated Diagnosis of Product-line Configuration Errors in Feature Models," Proceedings

of the Software Product Lines Conference (SPLC), September, 2008, Limerick, Ireland.

C141 Jules White and Douglas C. Schmidt, "Model-Driven Product-Line Architectures for Mobile Devices," Proceedings of the 17th Annual Conference of the International Federation of Automatic Control, Seoul, Korea, July 6-11, 2008.

C140 Jules White and Douglas C. Schmidt, "Automated Configuration of Component-based Distributed Real-time and Embedded Systems from Feature Models," Proceedings of the 17th Annual Conference of the International Federation of Automatic Control, Seoul, Korea, July 6-11, 2008.

C139 William R. Otte, John. S. Kinnebrew, Douglas C. Schmidt, Gautam Biswas, and Dipa Suri, "Application of Middleware and Agent Technologies to a Representative Sensor Network," Proceedings of the 2008 Earth Science Technology Conference, June 24-26, 2008, University of Maryland.

C138 Jai Balasubramanian, Aniruddha Gokhale, Douglas C. Schmidt, and Nanbor Wang, "Towards Middleware for Fault-tolerance in Distributed Real-time and Embedded Systems," Proceedings of the 8th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2008), Oslo, Norway, June 4-6, 2008.

C137 Serena Fritsch, Aline Senart, Douglas C. Schmidt, and Siobhan Clarke, "Time-bounded Adaptation for Automotive System Software," Proceedings of the Experience Track on Automotive Systems at the 30th International Conference on Software Engineering, Leipzig, Germany, 10 - 18 May 2008.

C136 Nilabja Roy, John S. Kinnebrew, Nishanth Shankaran, Gautam Biswas, and Douglas C. Schmidt, "Toward Effective Multi-capacity Resource Allocation in Distributed Real-time and Embedded Systems," Proceedings of the 11th IEEE International Symposium on Object/-Component/Service-oriented Real-time Distributed Computing, Orlando, Florida, May 5-7, 2008.

C135 Gan Deng, Douglas C. Schmidt, and Aniruddha Gokhale, "CaDANCE: A Criticality-Aware Deployment And Configuration Engine," Proceedings of the 11th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing, Orlando, Florida, May 5-7, 2008.

C134 Krishnakumar Balasubramanian and Douglas C. Schmidt, "Physical Assembly Mapper: A Model-driven Optimization Tool for QoS-enabled Component Middleware," Proceedings of 14th IEEE Real-Time and Embedded Technology and Applications Symposium, St. Louis, MO, United States, April 22 - April 24, 2008.

C133 Jaiganesh Balasubramanian, Sumant Tambe, Balakrishnan Dasarathy, Shrirang Gadgil, Frederick Porter, Aniruddha Gokhale, and Douglas C. Schmidt, "NetQoPE: A Model-driven Network QoS Provisioning Engine for Distributed Real-time and Embedded Systems," Proceedings of 14th IEEE Real-Time and Embedded Technology and Applications Symposium, St. Louis, MO, United States, April 22 - April 24, 2008.

C132 James Hill, Douglas C. Schmidt, John Slaby, and Adam Porter, "CiCUTS: Combining System Execution Modeling Tools with Continuous Integration Environments," Proceedings of the 15th Annual IEEE International Conference and Workshops on the Engineering of Computer Based Systems (ECBS), March 31st - 4th April, 2008 Belfast, Northern Ireland.

C131 Vinny Cahill, Aline Senart, Douglas C. Schmidt, Stefan Weber, Anthony Harrington, Barbara Hughes, and Kulpreet Singh, "The Managed Motorway: Real-time Vehicle Scheduling: A Research Agenda," Proceedings of the ACM HotMobile 2008, Silverado Resort, Napa Valley, CA, USA February 25-26, 2008.

C130 Jules White, Krzysztof Czarnecki, Douglas Schmidt, Gunther Lenz, Christoph Wienands, Egon Wuchner, Ludger Fiege, "Automated Model-based Configuration of Enterprise Java Applications," The 11th IEEE International EDOC Conference (EDOC 2007), 15-19 October 2007, Annapolis, Maryland U.S.A.

C129 Joe Hoffert, Douglas Schmidt, and Aniruddha Gokhale, "A QoS Policy Configuration Modeling Language for Publish/Subscribe Middleware Platforms," Proceedings of International Conference on Distributed Event-Based Systems (DEBS), June 20-22nd, 2007, Toronto, Canada.

C128 Shanshan Jiang, Yuan Xue, and Douglas Schmidt, "Minimum Disruption Service Composition and Recovery in Mobile Ad hoc Networks, Proceedings of the 4th Annual International

Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS 2007), August 6-10, 2007 - Philadelphia, PA.

C127 Jules White, Andrey Nechypurenko, Egon Wuchner, and Douglas C. Schmidt, "Optimizing and Automating Product-Line Variant Selection for Mobile Devices," Proceedings of the 11th International Software Product Line Conference, Kyoto, Japan, Sept 10-14, 2007.

C126 Nishanth Shankaran, Douglas C. Schmidt, Yingming Chen, Xenofon Koutsoukous, and Chenyang Lu, The Design and Performance of Configurable Component Middleware for End-to-End Adaptation of Distributed Real-time Embedded Systems, proceedings of the 10th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing, May 7-9, 2007, Santorini Island, Greece.

C125 Amogh Kavimandan, Krishnakumar Balasubramanian, Nishanth Shankaran, Aniruddha Gokhale, and Douglas C. Schmidt, QUICKER: A Model-driven QoS Mapping Tool, proceedings of the 10th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC), May 7-9, 2007, Santorini Island, Greece.

C124 Krishnakumar Balasubramanian, Douglas C. Schmidt, Zoltan Molnar, and Akos Ledeczi, Component-based System Integration via (Meta)Model Composition, Proceedings of the 14th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS), March 26th-29th, 2007, Tucson, Arizona.

C123 Joe Hoffert, Shanshan Jiang, and Douglas C. Schmidt, "A Taxonomy of Discovery Services and Gap Analysis for Ultra-Large Scale Systems," Proceedings of the ACMSE 2007: 45th ACM Southeast Conference Winston-Salem, North Carolina, USA March 23-24, 2007.

C122 Andrey Nechypurenko, Egon Wuchner, Jules White, and Douglas C. Schmidt, Application of Aspect-based Modeling and Weaving for Complexity Reduction in Development of Automotive Distributed Realtime Embedded System, Proceedings of the Sixth ACM International Conference on Aspect-Oriented Software Development, Vancouver, British Columbia, March 12-16, 2007.

C121 Dipa Suri, Adam Howell, Douglas C. Schmidt, Gautam Biswas, John Kinnebrew, Will Otte, and Nishanth Shankaran, "A Multi-agent Architecture for Smart Sensing in the NASA Sensor Web," Proceedings of the 2007 IEEE Aerospace Conference Big Sky, Montana, March 3-10, 2007.

C120 John S. Kinnebrew, Ankit Gupta, Nishanth Shankaran, Gautam Biswas, and Douglas C. Schmidt, A Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-Time Applications, Proceedings of the The 8th International Symposium on Autonomous Decentralized Systems (ISADS 2007) Sedona, Arizona, Wednesday March 21 - Friday March 23, 2007.

C119 Ming Xiong, Jeff Parsons, James Edmondson, and Douglas C. Schmidt, "Evaluating Technologies for Tactical Information Management in Net-Centric Systems, Proceedings of the Defense Transformation and Net-Centric Systems conference, April 9-13, 2007, Orlando, Florida.

C118 Nilabja Roy, Nishanth Shankaran, and Douglas C. Schmidt "Target Manager: A Resource Provisioning Service for Enterprise Distributed Real-time and Embedded Systems," Proceedings of the International Symposium on Distributed Objects and Applications (DOA), Montpellier, France, Oct 29 - Nov 3, 2006.

C117 James Hill, John Slaby, Steve Baker, and Douglas C. Schmidt, "Evaluating Enterprise Distributed Real-time and Embedded System Quality of Service with System Execution Modeling Tools," Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, Sydney, Australia, 16-18 August 2006.

C116 Nishanth Shankaran, Xenofon Koutsoukos, Chenyang Lu, Douglas C. Schmidt, and Yuan Xue, "Hierarchical Control of Multiple Resources in Distributed Real-time and Embedded Systems," Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS 06), Dresden, Germany, July 5-7, 2006.

C115 Dipa Suri, Adam Howell, Nishanth Shankaran, John Kinnebrew, Will Otte, Doug Schmidt, and Gautam Biswas, "Onboard Processing using the Adaptive Network Architecture", Proceedings of the Sixth annual NASA Earth Science Technology Conference MD, June 27th - 29th, 2006, College Park, MD.

C114 Gan Deng, Douglas C. Schmidt, Aniruddha Gokhale, and Andrey Nechypurenko, "Modularizing Variability and Scalability Concerns in Distributed Real-time and Embedded Systems

with Modeling Tools and Component Middleware, Proceedings of the 9th IEEE International Symposium on Object-oriented Real-time Distributed Computing (ISORC '06), April 24-26, 2006, Gyeongju, Korea.

C113 Stoyan Paunov and Douglas C. Schmidt, "RepoMan: A Component Repository Manager for Enterprise Distributed Real-time and Embedded Systems", Proceedings of the 44th ACM Southeast Conference, Melbourne, FL, March 10-12, 2006.

C112 Arvind S. Krishna, Aniruddha Gokhale, Douglas C. Schmidt, John Hatcliff, and Venkatesh Prasad Ranganat, "Context-Specific Middleware Specialization Techniques for Optimizing Software Product-line Architectures," Proceedings of ACM EuroSys 2006, Leuven, Belgium, April 18-21, 2006.

C111 Stoyan Paunov, James Hill, Douglas C. Schmidt, John Slaby, and Steve Baker, "Domain-Specific Modeling Languages for Configuring and Evaluating Enterprise DRE System Quality of Service," Proceedings of the 13th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS '06), March 27th-30th, 2006, University of Potsdam, Potsdam, Germany.

C110 Arvind S. Krishna, Aniruddha Gokhale, Douglas C. Schmidt, John Hatcliff, and Venkatesh Prasad Ranganat, "Towards Highly Optimized Real-time Middleware for Software Product-line Architectures," Proceedings of the Work-In-Progress session at the 26th IEEE Real-Time Systems Symposium, December 5-8, 2005, Miami, Florida.

C109 Gan Deng, Jaiganesh Balasubramanian, William Otte, Douglas C. Schmidt, and Aniruddha Gokhale, DAnCE: A QoS-enabled Component Deployment and Configuration Engine, Proceedings of the 3rd Working Conference on Component Deployment, Grenoble, France, November 28-29, 2005, p. 67-82.

C108 Jaiganesh Balasubramanian, Balachandran Natarajan, Douglas C. Schmidt, Aniruddha Gokhale, Gan Deng, and Jeff Parsons, "Evaluating Techniques for Dynamic Component Updating," Proceedings of the International Symposium on Distributed Objects and Applications (DOA), Agia Napa, Cyprus, Oct 31 - Nov 4, 2005, p. 978-996.

C107 Jules White, Douglas Schmidt, and Aniruddha Gokhale, "Simplifying Autonomic Enterprise Java Bean Applications via Model-driven Development: a Case Study," Proceedings of MODELS 2005, ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems, Half Moon Resort, Montego Bay, Jamaica, October 5-7, 2005, p. 601-615.

C106 Arvind Krishna, Douglas C. Schmidt, and Michael Stal, "Context Object A Design Pattern for Efficient Middleware Request Processing," Proceedings of the 12th Pattern Language of Programming Conference, Allerton Park, Illinois, September 7-10, 2005.

C105 Michael Stal and Douglas C. Schmidt, "Activator," Proceedings of the 12th Pattern Language of Programming Conference, Allerton Park, Illinois, September 7-10, 2005.

C104 Jules White, Boris Kolpackov, Balachandran Natarajan, and Douglas C. Schmidt, "Reducing Code Complexity With Vocabulary-Specific XML Language Bindings," Proceedings of the 43nd ACM Southeastern conference in Atlanta, GA, March 2005.

C103 Cemal Yilmaz, Arvind Krishna, Atif Memon, Adam Porter, Douglas C. Schmidt, Aniruddha Gokhale, and Bala Natarajan, "A Distributed Continuous Quality Assurance Process for Monitoring Performance Degradation in Evolving Software Systems, proceedings of the 27th International Conference on Software Engineering, St. Louis, MO, May 15-21, 2005, p. 293-302.

C102 Arvind S. Krishna, Emre Turkay, Aniruddha Gokhale, and Douglas C. Schmidt, Model-Driven Techniques for Evaluating the QoS of Middleware Configurations for DRE Systems, Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium, San Francisco, CA, March 2005, p. 180-189.

C101 Krishnakumar Balasubramanian, Jaiganesh Balasubramanian, Jeff Parsons, Aniruddha Gokhale, and Douglas C. Schmidt, "A Platform-Independent Component Modeling Language for Distributed Real-time and Embedded Systems," Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium, San Francisco, CA, March 2005, p. 190-199.

C100 Nanbor Wang, Christopher Gill, Douglas C. Schmidt, and Venkita Subramonian, "Configuring Real-time Aspects in Component Middleware," Proceedings of the Conference on Distributed Objects and Applications (DOA 2004), October 25-29, 2004, Cyprus, Greece.

C99 Jaiganesh Balasubramanian, Douglas C. Schmidt, Lawrence Dowdy, and Ossama Othman, "Evaluating the Performance of Middleware Load Balancing Strategies," Proceedings of the 8th International IEEE Enterprise Distributed Object Computing Conference, Monterey, California, September 20-24, 2004.

C98 George Edwards, Gan Deng, Douglas C. Schmidt, Aniruddha Gokhale, and Balachandran Natarajan, "Model-driven Configuration and Deployment of Component Middleware Publisher/Subscriber Services," Proceedings of the 3rd ACM International Conference on Generative Programming and Component Engineering, Vancouver, CA, October 2004.

C97 Andrey Nechypurenko, Douglas C. Schmidt, Tao Lu, Gan Deng, Emre Turkey, and Aniruddha Gokhale, "Concern-based Composition and Reuse of Distributed Systems," Proceedings of the 8th International Conference on Software Reuse, ACM/IEEE, Madrid, Spain, July 2004.

C96 Arvind Krishna, Douglas C. Schmidt, Adam Porter, Atif Memon, Diego Sevilla-Ruiz, "Improving the Quality of Performance-intensive Software via Model-integrated Distributed Continuous Quality Assurance," Proceedings of the 8th International Conference on Software Reuse, ACM/IEEE, Madrid, Spain, July 2004.

C95 Chris Gill, Jeanna M. Gossett, David Corman, Joseph P. Loyall, Richard E. Schantz, Michael Atighetchi, and Douglas C. Schmidt, "Integrated Adaptive QoS Management in Middleware: An Empirical Case Study," Proceedings of the 10th Real-time Technology and Application Symposium, May 25-28, 2004, Toronto, CA.

C94 Pradeep Gore, Douglas C. Schmidt, Chris Gill, and Irfan Pyarali, "The Design and Performance of a Real-time Notification Service," Proceedings of the 10th IEEE Real-time Technology and Application Symposium (RTAS '04), Toronto, CA, May 2004.

C93 Yamuna Krishnamurthy, Chris Gill, Douglas C. Schmidt, Irfan Pyarali, Louis Mgeta, Yuanfang Zhang, and Stephen Torri, "The Design and Performance of Real-time CORBA 2.0: Dynamic Scheduling in TAO," Proceedings of the 10th IEEE Real-time Technology and Application Symposium (RTAS '04), Toronto, CA, May 2004.

C92 Arvind S. Krishna, Nanbor Wang, Balachandran Natarajan, Aniruddha Gokhale, Douglas C. Schmidt and Gautam Thaker, "CCMPerf: A Benchmarking Tool for CORBA Component Model Implementations," Proceedings of the 10th IEEE Real-time Technology and Application Symposium (RTAS '04), Toronto, CA, May 2004.

C91 George Edwards, Douglas C. Schmidt, Aniruddha Gokhale, and Bala Natarajan, Integrating Publisher/Subscriber Services in Component Middleware for Distributed Real-time and Embedded Systems, Proceedings of the 42nd ACM Southeastern conference in Huntsville, AL, April 2004.

C90 Atif Memon, Adam Porter, Cemal Yilmaz, Adithya Nagarajan, Douglas C. Schmidt, and Bala Natarajan, "Skoll: Distributed Continuous Quality Assurance," Proceedings of the 26th IEEE/ACM International Conference on Software Engineering, Edinburgh, Scotland, May 2004.

C89 Arvind Krishna, Douglas C. Schmidt, and Raymond Klefstad, "Enhancing Real-Time CORBA via Real-Time Java," Proceedings of the 24th IEEE International Conference on Distributed Computing Systems (ICDCS), March 23-26, 2004, Tokyo, Japan.

C88 Arvind Krishna, Douglas C. Schmidt, Krishna Raman, and Raymond Klefstad, "Enhancing Real-time CORBA Predictability and Performance," Proceedings of the 5th International Symposium on Distributed Objects and Applications (DOA), Catania, Sicily, November 2003.

C87 Jeff Gray, Ted Bapty, Sandeep Neema, Douglas C. Schmidt, Aniruddha Gokhale, and Balachandran Natarajan, "An Approach for Supporting Aspect-Oriented Domain Modeling, Proceedings of the 2nd Generative Programming and Component Engineering (GPCE '03) conference, Erfurt, Germany, September 22-25, 2003.

C86 Richard E. Schantz, Joseph P. Loyall, Douglas C. Schmidt, Craig Rodrigues, Yamuna Krishnamurthy, and Irfan Pyarali, "Flexible and Adaptive QoS Control for Distributed Real-time and Embedded Middleware," Proceedings of Middleware 2003, 4th IFIP/ACM/USENIX International Conference on Distributed Systems Platforms, June 16-20, 2003, Rio de Janeiro, Brazil.

C85 Arvind Krishna, Douglas C. Schmidt, Raymond Klefstad, and Angelo Corsaro, "Towards Predictable Real-time Java Object Request Brokers," Proceedings of the 9th IEEE Real-

time/Embedded Technology and Applications Symposium (RTAS), Washington DC, May 28-30, 2003.

C84 Douglas C. Schmidt and Frank Buschmann, "Patterns, Frameworks, and Middleware: Their Synergistic Relationships," Proceedings of the IEEE/ACM International Conference on Software Engineering, Portland, Oregon, May 3-10, 2003.

C83 Radu Cornea, Nikil Dutt, Rajesh Gupta, Ingolf Krueger, Alex Nicolau, Douglas C. Schmidt, and Sandeep Shukla, "FORGE: A Framework for Optimization of Distributed Embedded Systems Software," International Parallel and Distributed Processing Symposium, Nice, France, April 22-26 2003.

C82 Raymond Klefstad, Sumita Rao, and Douglas C. Schmidt, "Design and Performance of a Dynamically Configurable, Messaging Protocols Framework for Real-time CORBA," Proceedings of the Distributed Object and Component-based Software Systems part of the Software Technology Track at the 36th Annual Hawaii International Conference on System Sciences, January 6–9, 2003, Big Island of Hawaii.

C81 Raymond Klefstad, Arvind S. Krishna, and Douglas C. Schmidt, "Design and Performance of a Modular Portable Object Adapter for Distributed, Real-Time, Embedded CORBA Applications," Proceedings of the Distributed Objects and Applications (DOA) conference, Irvine, CA, October/November, 2002.

C80 Chris Gill, Fred Kuhns, Douglas C. Schmidt, and Ron Cytron, "Empirical Differences Between COTS Middleware Scheduling Paradigms," Proceedings of the Distributed Objects and Applications (DOA) conference, Irvine, CA, October/November, 2002.

C79 Angelo Corsaro and Douglas C. Schmidt, "jRate: The Chameleonic Real-Time Java Implementation," Proceedings of the Distributed Objects and Applications (DOA) conference, Irvine, CA, October/November, 2002.

C78 Mayur Deshpande, Douglas C. Schmidt, Carlos O'Ryan, and Darrell Brunsch, "The Design and Performance of Asynchronous Method Handling for CORBA," Proceedings of the Distributed Objects and Applications (DOA) conference, Irvine, CA, October/November, 2002.

C77 Irfan Pyarali, Douglas C. Schmidt, and Ron Cytron, "Achieving End-to-End Predictability of the TAO Real-time CORBA ORB," Proceedings of the 8^{th} IEEE Real-Time Technology and Applications Symposium, San Jose, CA, September 2002.

C76 Angelo Corsaro and Douglas C. Schmidt, "Evaluating Real-Time Java Features and Performance for Real-time Embedded Systems," Proceedings of the 8^{th} IEEE Real-Time Technology and Applications Symposium, San Jose, CA, September 2002.

C75 Angelo Corsaro, Douglas C. Schmidt, Raymond Klefstad, and Carlos O'Ryan, "Virtual Component: a Design Pattern for Memory-Constrained Embedded Applications," Proceedings of the 9^{th} Annual Conference on the Pattern Languages of Programs, Monticello, Illinois, September, 2002.

C74 Joseph K. Cross and Douglas C. Schmidt, "Quality Connector: A Pattern Language for Provisioning and Managing Quality-Constrained Services in Distributed Real-time and Embedded Systems Proceedings of the 9th Annual Conference on the Pattern Languages of Programs, Monticello, Illinois, September, 2002.

C73 Richard Schantz, Franklin Webber, Partha Pal, Joseph Loyall, and Douglas C. Schmidt, "Protecting Applications Against Malice with Adaptive Middleware," Certification and Security in E-Services stream of the 17th IFIP World Computer Congress, Montreal, Canada, August 25-30, 2002.

c72 Richard Schantz and Douglas C. Schmidt, "Research Advances in Middleware for Distributed Systems: State of the Art," Computer Communications stream of the 17th IFIP World Computer Congress, Montreal, Canada, August 25-30, 2002.

C71 Raymond Klefstad, Douglas C. Schmidt, and Carlos O'Ryan, "Towards Highly Configurable Real-time Object Request Brokers," the IEEE International Symposium on Object-Oriented Real-time Distributed Computing (ISORC), Washington DC, April 29–May 1, 2002.

C70 Angelo Corsaro, Douglas C. Schmidt, Chris Gill, and Ron Cytron, "Formalizing Meta-Programming Techniques to Reconcile Heterogeneous Scheduling Policies in Open Distributed Real-Time Systems," Proceedings of the 3rd International Symposium on Distributed Objects and Applications, September 8-10, 2001, Rome, Italy.

C69 David A. Karr, Craig Rodrigues, Yamuna Krishnamurthy, Irfan Pyarali, and Douglas C. Schmidt "Application of the QuO Quality-of-Service Framework to a Distributed Video Application," Proceedings of the 3rd International Symposium on Distributed Objects and Applications, September 8-10, 2001, Rome, Italy.

C68 Nanbor Wang, Kirthika Parameswaran, and Douglas C. Schmidt, "The Design and Performance of Meta-Programming Mechanisms for Object Request Broker Middleware," Proceedings of the 6th USENIX Conference on Object-Oriented Technologies and Systems (COOTS), San Antonio, TX, Jan/Feb, 2001.

C67 Andy Gokhale, Bala Natarajan, Douglas C. Schmidt and Shalini Yajnik, "Applying Patterns to Improve the Performance of Fault-Tolerant CORBA," of the 7th International Conference on High Performance Computing (HiPC 2000), ACM/IEEE, Bangalore, India, December 2000.

C66 Nanbor Wang, Michael Kircher, and Douglas C. Schmidt, "Applying Reflective Techniques to Optimize a QoS-enabled CORBA Component Model Implementation, the 24th Annual International Computer Software and Applications Conference (COMPSAC 2000), Taipai, Taiwan, October 25-27 2000.

C65 Frederic Andres, Nicolas Dessaigne, Jose Martinez, Noureddine Mouaddib, Kinji Ono, Douglas C. Schmidt, Panrit Tosukhowong, "MISE: The MediaSys Image Search Engine," 11th International Conference on Database and Expert Systems Applications (DEXA 2000), London, UK, September 2000.

C64 Andy Gokhale, Bala Natarajan, Douglas C. Schmidt and Shalini Yajnik, "DOORS: Towards High-performance Fault-Tolerant CORBA," Proceedings of the 2nd International Symposium on Distributed Objects and Applications (DOA '00), OMG, Antwerp, Belgium, September 2000.

C63 Irfan Pyarali, Carlos O'Ryan, Douglas C. Schmidt, "A Pattern Language for Efficient, Predictable, Scalable, and Flexible Dispatching Components," Proceedings of the 7th Pattern Language of Programming Conference, Monticello, Illinois, August, 2000.

C62 Douglas C. Schmidt, Carlos O'Ryan, Irfan Pyarali, Michael Kircher and Frank Buschmann, "Leader/Followers: A Design Pattern for Efficient Multi-threaded Event Demultiplexing and Dispatching," Proceedings of the 7th Pattern Languages of Programming Conference, Monticello, Illinois, August 2000.

C61 Carlos O'Ryan, Douglas C. Schmidt, Fred Kuhns, Marina Spivak, Jeff Parsons Irfan Pyarali, and David L. Levine, "Evaluating Policies and Mechanisms for Supporting Embedded, Real-Time Applications with CORBA 3.0," Proceedings of the Sixth IEEE Real-Time Technology and Applications Symposium (RTAS'00), Washington D.C., USA, May 31-June 2, 2000.

C60 Nanbor Wang, Douglas C. Schmidt, and David Levine, "Optimizing the CORBA Component Model for High-performance and Real-time Applications," Work-in-progress session of the IFIP/ACM Middleware 2000 Conference, Pallisades, New York, April 3-7, 2000.

C59 Alexander B. Arulanthu, Carlos O'Ryan, Douglas C. Schmidt, Michael Kircher, and Jeff Parsons, "The Design and Performance of a Scalable ORB Architecture for CORBA Asynchronous Messaging," Proceedings of the IFIP/ACM Middleware 2000 Conference, Pallisades, New York, April 3-7, 2000.

C58 Carlos O'Ryan, Fred Kuhns, Douglas C. Schmidt, Ossama Othman, and Jeff Parsons, The Design and Performance of a Pluggable Protocols Framework for Real-time Distributed Object Computing Middleware, Proceedings of the IFIP/ACM Middleware 2000 Conference, Pallisades, New York, April 3-7, 2000.

C57 Irfan Pyarali, Carlos O'Ryan, and Douglas C. Schmidt, "A Pattern Language for Efficient, Predictable, Scalable, and Flexible Dispatching Mechanisms for Distributed Object Computing Middleware," Proceedings of the IEEE/IFIP International Symposium on Object-Oriented Real-time Distributed Computing, March 15-17, 2000, Newport Beach, California.

C56 David Levine, Douglas C. Schmidt, and Sergio Flores-Gaitan, "An Empirical Evaluation of OS Support for Real-time CORBA Object Request Brokers," Proceedings of the Multimedia Computing and Networking 2000 (MMCN00) conference, ACM, San Jose, CA, January 25-27 2000.

C55 Douglas C. Schmidt, "Middleware Techniques and Optimizations for Real-time, Embedded Systems," Proceedings of the 12th International Symposium On System Synthesis, IEEE, San Jose, CA, USA November, 11, 1999.

C54 Panrit Tosukhowong, Frederic Andres, Kinji Ono, Nicolas Dessaigne, Josi Martinez, Nouredine Mouaddib, Douglas C. Schmidt, “A Flexible Image Search Engine,” Proceedings of International Multimedia Conference Archive Proceedings of the Seventh ACM International Conference on Multimedia, Orlando, Florida, United States, November 1-5, 1999.

C53 Bryan S. Doerr, Thomas Venturella, Rakesh Jha, Christopher D. Gill, and Douglas C. Schmidt, “Adaptive Scheduling for Real-time, Embedded Information Systems,” Proceedings of the 18th IEEE/AIAA Digital Avionics Systems Conference (DASC), St. Louis, Missouri, October 24-29, 1999.

C52 Christopher D. Gill, David L. Levine, Carlos O’Ryan, and Douglas C. Schmidt, “Distributed Object Visualization for Sensor-Driven Systems,” Proceedings of the 18th IEEE/AIAA Digital Avionics Systems Conference (DASC), St. Louis, Missouri, October 24-29, 1999.

C51 Fred Kuhns, Douglas C. Schmidt, and David L. Levine, “The Performance of a Real-time I/O Subsystem for QoS-enabled ORB Middleware,” Proceedings of the International Symposium on Distributed Objects and Applications (DOA ’99), Edinburgh, Scotland, September 1999.

C50 David L. Levine, Christopher D. Gill, and Douglas C. Schmidt, “Object Lifecycle Manager – A Complementary Pattern for Controlling Object Creation and Destruction.” Proceedings of the 5th Pattern Languages of Programing Conference, Allerton Park, Illinois, USA, 15–18 August 1999.

C49 Fred Kuhns, Douglas C. Schmidt, David Levine, and Rajeev Bector, “The Design and Performance of a Real-time I/O Subsystem,” Proceedings of the 5th IEEE Real-Time Technology and Applications Symposium (RTAS99), Vancouver, British Columbia, Canada, June 2-4, 1999.

C48 Irfan Pyarali, Carlos O’Ryan, Douglas C. Schmidt, Nanbor Wang, Vishal Kachroo, and Aniruddha Gokhale, “Applying Optimization Patterns to Design Real-time ORBs,” Proceedings of the 5th USENIX Conference on Object-Oriented Technologies and Systems, May 3-7, 1999, San Diego, CA.

C47 Andy Gokhale and Douglas C. Schmidt, “Techniques for Optimizing CORBA Middleware for Distributed Embedded Systems” Proceedings of INFOCOM ’99, March 21-25th, New York, New York.

C46 Sumedh Mungee, Nagarajan Surendran, and Douglas C. Schmidt, “The Design and Performance of a CORBA Audio/Video Streaming Service,” Proceedings of the 31st Hawaii International Conference on System Systems (HICSS), Hawaii, January, 1999, minitrack on Multimedia DBMS and the WWW, Hawaii, January 1999.

C45 Chris D. Gill, David L. Levine, and Douglas C. Schmidt, “Dynamic Scheduling for Avionics Applications,” Proceedings of the 17th IEEE/AIAA Digital Avionics System Conference, 31 October - 6 November 1998.

C44 James Hu, Irfan Pyarali, and Douglas C. Schmidt, “Applying the Proactor Pattern to High-Performance Web Servers,” Proceedings of the 10th International Conference on Parallel and Distributed Computing and Systems, IASTED, Las Vegas, Nevada, October 28-31, 1998.

C43 Douglas C. Schmidt, Sumedh Mungee, and Andy Gokhale, “Alleviating Priority Inversion and Non-determinism in Real-time CORBA ORB Core Architectures,” Proceedings of the Fourth IEEE Real-Time Technology and Applications Symposium (RTAS), Denver, Colorado, June 3-5, 1998.

C42 Prashant Jain, Seth Widoff, and Douglas C. Schmidt, “The Design and Performance of Med-Java, A Distributed Electronic Medical Imaging System Developed with Java Applets and Web Tools” Proceedings of the 4th USENIX Conference on Object-Oriented Technologies and Systems, Sante Fe, New Mexico, April 1998. This was selected as the best student paper in the conference.

C41 James Hu, Sumedh Mungee, and Douglas C. Schmidt, “Techniques for Developing and Measuring High-performance Web Servers over ATM Networks,” Proceedings of INFOCOM ’98, San Francisco, March/April, 1998.

C40 Aniruddha Gokhale and Douglas C. Schmidt, “Optimizing the Performance of the CORBA Internet Inter-ORB Protocol Over ATM,” Proceedings of the 31st Hawaii International Conference on System Systems (HICSS), Hawaii, January, 1998. This was selected as the best paper in the Software Technology Track (188 submitted, 77 accepted).

C39 Aniruddha Gokhale and Douglas C. Schmidt, "Evaluating the Performance of Demultiplexing Strategies for Real-time CORBA," Proceedings of GLOBECOM '97 conference, IEEE, Phoenix, AZ, November, 1997.

C38 James Hu, Irfan Pyarali, and Douglas C. Schmidt, "Measuring the Impact of Event Dispatching and Concurrency Models on Web Server Performance Over High-speed Networks," Proceedings of the 2nd Global Internet Conference (held as part of GLOBECOM '97) in Phoenix, AZ, November 4-8, 1997.

C37 Tim Harrison and David Levine and Douglas C. Schmidt, "The Design and Performance of a Real-time CORBA Event Service," Proceedings of OOPSLA '97, ACM, Atlanta, GA, October 1997.

C36 Aniruddha Gokhale and Douglas C. Schmidt and Stan Moyer, "Tools for Automating the Migration from DCE to CORBA,"Proceedings of ISS 97: World Telecommunications Congress, IEEE Toronto, Canada, September, 1997.

C35 Jain and Douglas C. Schmidt, "Service Configurator – A Pattern for Dynamic Configuration of Services," the 4th annual Pattern Languages of Programmingy conference in Allerton Park, Illinois, September 1997.

C34 Chris Cleeland, Douglas C. Schmidt, and Tim H. Harrison, "External Polymorphism – An Object Structural Pattern for Transparently Extending C++ Concrete Data Types," the 4th annual Pattern Languages of Programmingy conference in Allerton Park, Illinois, September 1997.

C33 Douglas C. Schmidt, Tim H. Harrison, and Nat Pryce, "Thread-specific Storage: an Object Behavioral Pattern for Efficiently Accessing per-Thread State," The 4th annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 1997.

C32 Irfan Pyarali, Tim Harrison, Douglas C. Schmidt, and Thomas Jordan, "Proactor: an Object Behavioral Pattern for Demultiplexing and Dispatching Handlers for Asynchronous Events," the 4th annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 1997.

C31 Prashant Jain and Douglas C. Schmidt, "Service Configurator – A Pattern for Dynamic Configuration of Services," Proceedings of the 3rd Conference on Object-Oriented Technologies and Systems, USENIX, Portland, OR, June 16-19, 1997.

C30 Aniruddha Gokhale and Douglas C. Schmidt, "Evaluating Latency and Scalability of CORBA Over High-Speed ATM Networks," Proceedings of the International Conference on Distributed Computing Systems '97, IEEE, Baltimore, Maryland, May 27–30, 1997.

C29 Aniruddha Gokhale and Douglas C. Schmidt, "Performance of the CORBA Dynamic Invocation Interface and Internet Inter-ORB Protocol over High-Speed ATM Networks," Proceedings of GLOBECOM '96, IEEE, London England, November, 1996.

C28 Aniruddha Gokhale and Douglas C. Schmidt, "Measuring the Performance of Communication Middleware on High-Speed Networks," Proceedings of SIGCOMM '96, ACM, San Francisco, August 28-30th, 1996.

C27 Irfan Pyarali, Tim Harrison, and Douglas. C. Schmidt, "Design and Performance of an Object-Oriented Framework for High-Speed Electronic Medical Imaging," Proceedings of the 2nd Conference on Object-Oriented Technologies and Systems (COOTS), USENIX, Toronto, June 18-22, 1996.

C26 Douglas C. Schmidt, "A Family of Design Patterns For Flexibly Configuring Network Services in Distributed Systems," Proceedings of the International Conference on Configurable Distributed Systems, IEEE, Annapolis, Maryland, May 6-8, 1996.

C25 Douglas C. Schmidt "Using Design Patterns to Develop High-Performance Object-Oriented Communication Software Frameworks," Proceedings of the 8th Annual Software Technology Conference, Salt Lake City, Utah, April 21-26, 1996.

C24 Douglas C. Schmidt, Timothy H. Harrison, and Irfan Pyarali, "An Object-Oriented Framework for High-Performance Electronic Medical Imaging," Proceedings of the *Very High Resolution and Quality Imaging* mini-conference at the Symposium on Electronic Imaging in the International Symposia Photonics West 1996, SPIE, San Jose, California USA, January 27 - February 2, 1996.

C23 Douglas C. Schmidt and Charles D. Cranor, "Half-Sync/Half-Async: A Pattern for Efficient and Well-structured Concurrent I/O," *The 2nd Pattern Languages of Programs Conference* Monticello, Illinois, September 6-8, 1995.

C22 R. Greg Lavender and Douglas C. Schmidt, "Active Object: An Object Behavioral Pattern for Concurrent Programming," *The 2nd Pattern Languages of Programs Conference*, Monticello, Illinois, September 6-8, 1995.

C21 Guru Parulkar, Douglas C. Schmidt, and Jonathan S. Turner, "$_aI_tP_m$: a Strategy for Integrating IP with ATM," the Symposium on Communications Architectures and Protocols (SIGCOMM), ACM, Cambridge, MA, August 30 to September 1, 1995.

C20 Douglas C. Schmidt, Tim Harrison, and Ehab Al-Shaer, "Object-Oriented Components for High-speed Network Programming," *Proceedings of the Conference on Object-Oriented Technologies (COOTS)*, USENIX, June 26-29, 1995 Monterey, California, USA, pp. 21-38.

C19 Douglas C. Schmidt and Paul Stephenson, "Experience Using Design Patterns to Evolve Communication Software Across Diverse OS Platforms," *Proceedings of the 9th European Conference on Object-Oriented Programming (ECOOP)*, ACM, Aarhus, Denmark, August, 1995.

C18 Douglas C. Schmidt and Tatsuya Suda, "Measuring the Performance of Parallel Message-based Process Architectures," *Proceedings of the INFOCOM Conference on Computer Communications*, IEEE, Boston, MA, April, 1995, pp. 624-633.

C17 Douglas C. Schmidt and Tatsuya Suda, "Experiences with an Object-Oriented Architecture for Developing Dynamically Extensible Network Management Software," *Proceedings of the Globecom Conference*, IEEE, San Francisco, California, November, 1994, pp. 1-7.

C16 Douglas C. Schmidt and Paul Stephenson, "Achieving Reuse Through Design Patterns," *Proceedings of the 3rd Annual C++ World Conference*, SIGS, Austin, Texas, November 14-18, 1994.

C15 Douglas C. Schmidt, "Developing Object-Oriented Frameworks to Dynamically Configure Concurrent, Multi-service Network Daemons," *Proceedings of the 3rd Annual C++ World Conference*, SIGS, Austin, Texas, November 14-18, 1994.

C14 Douglas C. Schmidt, "Reactor: An Object Behavioral Pattern for Concurrent Event Demultiplexing and Dispatching," *The 1st Annual Conference on the Pattern Languages of Programs*, Monticello, Illinois, August, 1994, pp. 1-10.

C13 Douglas C. Schmidt, "The ADAPTIVE Communication Environment: An Object-Oriented Network Programming Toolkit for Developing Communication Software," *Proceedings of the 12th Annual Sun Users Group Conference*, SUG, San Francisco, June 16-17, 1994. This paper won the "best student paper" award at the conference.

C12 Douglas C. Schmidt, Burkhard Stiller, Tatsuya Suda, and Martina Zitterbart, "Configuring Function-based Communication Protocols for Distributed Applications," *Proceedings of the 8th International Working Conference on Upper Layer Protocols, Architectures, and Applications*, IFIP, Barcelona, Spain, June 1-3, 1994, pp. 361-376.

C11 Douglas C. Schmidt and Tatsuya Suda, "The ADAPTIVE Service Executive: An Object-Oriented Architecture for Configuring Concurrent Distributed Communication Systems," *Proceedings of the 8th International Working Conference on Upper Layer Protocols, Architectures, and Applications*, IFIP, Barcelona, Spain, June 1-3, 1994, pp. 163-178.

C10 Douglas C. Schmidt, "ASX: An Object-Oriented Framework for Developing Distributed Applications," *Proceedings of the 6th C++ Conference*, USENIX, Cambridge, Massachusetts, April, 1994, pp. 200-220.

C9 Douglas C. Schmidt, "The ADAPTIVE Communication Environment: Object-Oriented Network Programming Components for Developing Client/Server Applications," *Proceedings of the 11th Annual Sun Users Group Conference*, SUG, San Jose, December 7-9, 1993, pp. 214-225. This paper won the "best student paper" award at the conference.

C8 Douglas C. Schmidt and Paul Stephenson, "An Object-Oriented Framework for Developing Network Server Daemons," *Proceedings of the 2nd Annual C++ World Conference*, SIGS, Dallas, Texas, October 18-22, 1993, pp. 73-85.

C7 Douglas C. Schmidt, "Object-Oriented Techniques for Developing Extensible Network Servers," *Proceedings of the 2nd Annual C++ World Conference*, SIGS, Dallas, Texas, October 18-22, 1993.

C6 Douglas C. Schmidt, Burkhard Stiller, Tatsuya Suda, Ahmed Tantawy, and Martina Zitterbart, "Configuration Support for Flexible Function-Based Communication Systems," *Proceedings of the 18th Conference on Local Computer Networks*, IEEE, Minneapolis, Minnesota, September 20-22, 1993, pp. 369-378.

C5 Douglas C. Schmidt and Tatsuya Suda, "ADAPTIVE: a Framework for Experimenting with High-Performance Transport System Process Architectures," *Proceedings of the 2nd International Conference on Computer Communications and Networks*, ISCA, San Diego, California, June 28-30, 1993, pp. 1-8.

C4 Donald F. Box, Douglas C. Schmidt, and Tatsuya Suda, "ADAPTIVE: An Object-Oriented Framework for Flexible and Adaptive Communication Protocols," *Proceedings of the 4th Conference on High Performance Networking*, IFIP, Liege, Belgium, December 14-18, 1992, pp. 367-382.

C3 Douglas C. Schmidt, Donald F. Box, and Tatsuya Suda, "ADAPTIVE: A Flexible and Adaptive Transport System Architecture to Support Lightweight Protocols for Multimedia Applications on High-Speed Networks," *Proceedings of the 1st Symposium on High Performance Distributed Computing*, IEEE, Syracuse, New York, September 9-11, 1992, pp. 174-186.

C2 Richard W. Selby, Adam A. Porter, Douglas C. Schmidt, and James Berney, "Metric-Driven Analysis and Feedback Systems for Enabling Empirically Guided Software Development," *Proceedings of the 13th Annual International Conference on Software Engineering*, IEEE, Austin, Texas, May, 1991, pp. 430-443.

C1 Douglas C. Schmidt "GPERF: A Perfect Hash Function Generator," *Proceedings of the 2nd C++ Conference*, USENIX, San Francisco, California, April 9-11, 1990, pp. 87-102.

- **Refereed Workshop Publications**

W74 Nick Guertin, Douglas C. Schmidt, and Harry Levinson, "Business and Organizational Impacts for Modular Flexible Ships," Proceedings of the 2018 Design Sciences Series Workshop on Modular Adaptable Ships, Washington DC, November 14-15, 2018.

W73 Michael Walker, Abhishek Dubey, Aron Laszka, and Douglas C. Schmidt, "PlaTIBART: a Platform for Transactive IoT Blockchain Applications with Repeatable Testing," Proceedings of the ACM/IFIP/USENIX 4th Workshop on Middleware and Applications for the Internet of Things, December 2017, Las Vegas, USA.

W72 Abhishek Dubey, Subhav Pradhan, Douglas C. Schmidt, Sebnem Rusitschka, and Monika Sturm, "The Role of Context and Resilient Middleware in Next Generation Smart Grids," Proceedings of the 3rd Middleware for Context-Aware Applications in the IoT (M4IOT 2016) Workshop at the ACM/IFIP/USENIX Middleware 2016 Conference, Dec 12 - 16, 2016, Trento, Italy.

W71 Violetta Vylegzhanina, Douglas C. Schmidt, and Jules White, "Gaps and Future Directions in Mobile Security Research," Proceedings of the Third International Workshop on Mobile Development Lifecycle, Pittsburgh, PA, October 26th, 2015.

W70 Violetta Vylegzhanina, Douglas C. Schmidt, Pamela Hull, Janice S. Emerson, Meghan E. Quirk, and Shelagh Mulvaney, "Helping Children Eat Well Via Mobile Software Technologies," Proceedings of the Second International Workshop on Mobile Development Lifecycle, October 21st, 2015, Portland, OR.

W69 Jules White and Douglas C. Schmidt, "R&D Challenges and Emering Softwares for Multicore Deployment/Configuration Optimization," proceedings of the ACM Workshop on Future of Software Engineering Research (FoSER 2010), Santa Fe, NM, November 7-11, 2010.

W68 Will Otte, Douglas C. Schmidt, and Aniruddha Gokhale, "Towards an Adaptive Deployment and Configuration Framework for Component-based Distributed Systems," Proceedings of the 9th Workshop on Adaptive and Reflective Middleware (ARM 2010) November 27, 2010, Bangalore India, collocated with Middleware 2010.

W67 Jaiganesh Balasubramanian, Alexander Mintz, Andrew Kaplan, Grigory Vilkov, Artem Gleyzer, Antony Kaplan, Ron Guida, Pooja Varshneya and Douglas Schmidt, "Adaptive Parallel Computing for Large-scale Distributed and Parallel Applications," Proceedings of the Workshop on Data Dissemination for Large-scale Complex Critical Infrastructures (DD4LCCI), 27 April 2010, in conjunction with EDCC 2010, Valencia - Spain, April 28-30, 2010.

W66 Joe Hoffert, Douglas Schmidt, and Aniruddha Gokhale, "Adapting and Evaluating Distributed Real-time and Embedded Systems in Dynamic Environments," Proceedings of the Workshop on Data Dissemination for Large-scale Complex Critical Infrastructures (DD4LCCI), 27 April 2010, in conjunction with EDCC 2010, Valencia - Spain, April 28-30, 2010.

W65 Joe Hoffert, Dan Mack, and Douglas C. Schmidt, "Using Machine Learning to Maintain Pub/Sub System QoS in Dynamic Environments, Proceedings of the 8th Workshop on Adaptive and Reflective Middleware (ARM'09) December 1st 2009, Urbana Champaign, Illinois, USA collocated with Middleware 2009.

W64 Chris Thompson, Jules White, Brian Dougherty, and Douglas C. Schmidt, "Optimizing Mobile Application Performance with Model-Driven Engineering," Proceedings of the 7th IFIP Workshop on Software Technologies for Future Embedded and Ubiquitous Systems (SEUS 2009), November 16-18, 2009, Newport Beach, California.

W63 Jules White and Douglas C. Schmidt, "Filtered Cartesian Flattening: An Approximation Technique for Optimally Selecting Features while Adhering to Resource Constraints," proceedings of the Workshop on Analyses of Software Product Lines (ASPL 2008) at the Software Product Lines Conference (SPLC), September 8-12, 2008, Limerick, Ireland.

W62 Joe Hoffert, Douglas C. Schmidt, Mahesh Balakrishnan, and Ken Birman, Supporting Large-scale Continuous Stream Datacenters via Pub/Sub Middleware and Adaptive Transport Protocols, Proceedings of the 2nd workshop on Large-Scale Distributed Systems and Middleware (LADIS 2008), IBM TJ Watson Research Center, Yorktown, New York, September 2008.

W61 Nishanth Shankaran, John S. Kinnebrew, Xenofon D. Koutsoukos, Chenyang Lu, Douglas C. Schmidt, and Gautam Biswas, Towards an Integrated Planning and Adaptive Resource Management Architecture for Distributed Real-time Embedded Systems," Proceedings of the Workshop on Adaptive and Reconfigurable Embedded Systems (APRES) at the 14th IEEE Real-Time and Embedded Technology and Applications Symposium, St. Louis, MO, United States, April 22 - April 24, 2008.

W60 Serena Fritsch, Aline Senart, Douglas C. Schmidt, and Siobhan Clarke, "Scheduling Time-bounded Dynamic Software Adaptation," Proceedings of the workshop on Software Engineering for Adaptive and Self-Managing Systems at the 30th IEEE/ACM International Conference on Software Engineering May 12-13, 2008, Leipzig, Germany.

W59 James Hill, Jules White, Sean Eade, and Douglas C. Schmidt, "Towards a Solution for Synchronizing Disparate Models of Ultra-Large-Scale Systems," Proceedings of the Second International Workshop on Ultra-Large-Scale Software- Intensive Systems at the 30th IEEE/ACM International Conference on Software Engineering May 10-11, 2008, Leipzig, Germany.

W58 Douglas C. Schmidt and Hans van't Hag, "Addressing the Challenges of Tactical Information Management in Net-Centric Systems with OpenSplice DDS," Proceedings of the 16th International ACM Workshop on Parallel and Distributed Real-Time Systems (WPDRTS '08), Miami, Florida, April 2008.

W57 Shanshan Jiang, Yuan Xue, and Douglas C. Schmidt, "Disruption-Aware Service Composition and Recovery in Dynamic Networking Environments," WoRkshop on Automating Service Quality (WRAQS) 2007, Co-Located with ASE 2007 November 6, 2007, Atlanta, Georgia.

W56 Jules White, Douglas C. Schmidt, Sean Mulligan, "The Generic Eclipse Modeling System," Model-Driven Development Tool Implementer's Forum, TOOLS '07, June, 2007, Zurich, Switzerland.

W55 John S. Kinnebrew, Nishanth Shankaran, Gautam Biswas, and Douglas C. Schmidt, A Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-time and Embedded Systems, Proceedings of the Workshop on Artificial Intelligence for Space Applications at IJCAI 2007, January 6-12, 2007, Hyderabad, India.

W54 Andrey Nechypurenko, Jules White, Egon Wuchner, and Douglas C. Schmidt, "Applying Model Intelligence Frameworks for Deployment Problem in Real-time and Embedded Systems," Proceedings of MARTES: Modeling and Analysis of Real-Time and Embedded Systems to be held on October 2, 2006 in Genova, Italy in conjunction with the 9th International Conference on Model Driven Engineering Languages and Systems, MoDELS/UML 2006.

W53 Jules White, Andrey Nechypurenko, Egon Wuchner, and Douglas C. Schmidt, "Intelligence Frameworks for Assisting Modelers in Combinatorically Challenging Domains," Proceedings of

the Workshop on Generative Programming and Component Engineering for QoS Provisioning in Distributed Systems, October 23, 2006, Portland, Oregon.

W52 Nishanth Shankaran, Xenofon Koutsoukos, Douglas C. Schmidt, and Aniruddha Gokhale, "Evaluating Adaptive Resource Management for Distributed Real-Time Embedded Systems," Proceedings of the 4th Workshop on Adaptive and Reflective Middleware, November 28, 2005 Grenoble, France.

W51 Jules White and Douglas Schmidt, "Simplifying the Development of Product-line Customization Tools via Model Driven Development," MODELS 2005 workshop on MDD for Software Product-lines: Fact or Fiction?, October 2, 2005, Jamaica.

W50 Arvind S. Krishna, Aniruddha Gokhale, Douglas C. Schmidt, Venkatesh Prasad Ranganath, and John Hatcliff, "Model-driven Middleware Specialization Techniques for Software Product-line Architectures in Distributed Real-time and Embedded Systems," MODELS 2005 workshop on MDD for Software Product-lines: Fact or Fiction?, October 2, 2005, Jamaica.

W49 Gen Deng, Gunther Lenz, and Douglas C. Schmidt, "Addressing Domain Evolution Challenges in Model-Driven Software Product-line Architectures," MODELS 2005 workshop on MDD for Software Product-lines: Fact or Fiction?, October 2, 2005, Jamaica.

W48 Andrey Nechypurenko and Douglas C. Schmidt, "Supporting Model Reusability with Pattern-based Composition Units," Proceedings of the IST 2nd European Workshop on Model Driven Architecture (MDA), with an emphasis on Methodologies and Transformations September 7th-8th 2004, Canterbury, England.

W47 Cemal Yilmaz, Arvind S. Krishna, Atif Memon, Adam Porter, Douglas C. Schmidt, Aniruddha Gokhale, and Balachandran Natarajan, "A Model-based Distributed Continuous Quality Assurance Process to Enhance the Quality of Service of Evolving Performance-intensive Software Systems," Proceedings of the 2nd ICSE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS), Edinburgh, Scotland, UK, May 24, 2004.

W46 Andrey Nechypurenko, Douglas C. Schmidt, Tao Lu, Gan Deng, Aniruddha Gokhale, "Applying MDA and Component Middleware to Large-scale Distributed Systems: a Case Study, Proceedings of the OMG 1st European Workshop on Model Driven Architecture with Emphasis on Industrial Application, Enschede, the Netherlands, March 2004.

W45 Gan Deng, Tao Lu, Emre Turkay, Aniruddha Gokhale, Douglas C. Schmidt, and Andrey Nechypurenko, "Model Driven Development of Inventory Tracking System," Proceedings of the OOPSLA 2003 Workshop on Domain-Specific Modeling Languages, Anaheim, CA, October 2003.

W44 Tao Lu, Emre Turkay, Aniruddha Gokhale, and Douglas C. Schmidt, "CoSMIC: An MDA Tool suite for Application Deployment and Configuration," Proceedings of the OOPSLA 2003 Workshop on Generative Techniques in the Context of Model Driven Architecture, Anaheim, CA, October 2003.

W43 Arvind S. Krishna, Jai Balasubramanian, Aniruddha Gokhale, Douglas C. Schmidt, Diego Sevilla, and Gautham Thaker, "Empirically Evaluating CORBA Component Model Implementations," Proceedings of the ACM OOPSLA 2003 Workshop on Middleware Benchmarking, Anaheim, CA, October 2003.

W42 Aniruddha Gokhale, Douglas C. Schmidt, Tao Lu, Balachandran Natarjan, and Nanbor Wang, CoSMIC: An MDA Generative Tool for Distributed Real-time and Embedded Applications, Workshop on Model-driven Approaches to Middleware Applications Development at 4th IFIP/ACM/USENIX International Conference on Middleware for Distributed Systems Platforms, June 16, 2003, Rio de Janeiro, Brazil.

W41 Ossama Othman, Jaigaesh Balasubramanian, and Douglas C. Schmidt, "The Design and Performance of an Adaptive Middleware Load Balancing and Monitoring Service," Third International Workshop on Self-Adaptive Software, Arlington, VA, USA, June 9-11, 2003.

W40 Radu Cornea, Shivajit Mohapatra, Nikil Dutt, Rajesh Gupta, Ingolf Krueger, Alex Nicolau, Doug Schmidt, Sandeep Shukla, and Nalini Venkatasubramanian, "A Model-Based Approach to System Specification for Distributed Real-time and Embedded Systems," IEEE RTAS Workshop on Model-Driven Embedded Systems, Washington DC, May 27-30, 2003.

W39 Adam Porter, Cemal Yilmaz, and Douglas C. Schmidt "Distributed Continuous Quality Assurance: The Skoll Project," Proceedings of the Workshop on Remote Analysis and Measurement of Software Systems (RAMSS), Portland, Oregon, May 9, 2003.

W38 Krishnakumar Balasubramanian, Douglas C. Schmidt, Nanbor Wang, Christopher D. Gill, "Towards Composable Distributed Real-time and Embedded Software," Proceedings of the 8^{th} IEEE Workshop on Object-oriented Real-time Dependable Systems, Guadalajara, Mexico, January 2003.

W37 Aniruddha Gokhale, Balachandran Natarjan, Douglas C. Schmidt, Andrey Nechypurenko, Nanbor Wang, Jeff Gray, Sandeep Neema, Ted Bapty, and Jeff Parsons, "CoSMIC: An MDA Generative Tool for Distributed Real-time and Embedded Component Middleware and Applications," Proceedings of the OOPSLA 2002 Workshop on Generative Techniques in the Context of Model Driven Architecture, Seattle, WA, November 2002.

W36 M. Mousavi, G. Russello, M. Chaudron, M. Reniers, T. Basten, A. Corsaro, S. Shukla, R. Gupta, and D.C. Schmidt, "Using Aspect-GAMMA in Design and Verification of Embedded Systems," Proceedings of the Seventh Annual IEEE International Workshop on High Level Design Validation and Test Workshop, Cannes, France, October 27-29, 2002.

W35 Douglas C. Schmidt, Andy Gokhale, and Chris Gill, "Applying Model-Integrated Computing and DRE Middleware to High Performance Embedded Computing Applications," Proceedings of the 6th Annual Workshop on High-Performance Embedded Computing (HPEC), September 24-26, Boston, MA.

W34 Douglas C. Schmidt, "Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems," EMSOFT 2002: Second Workshop on Embedded Software, Grenoble, France, October, 7-9th, 2002.

W33 Aniruddha S. Gokhale and Douglas C. Schmidt and Joseph K. Cross and Christopher Andrews and Sylvester J. Fernandez and Bala Natarajan and Nanbor Wang and Chris D. Gill, "Towards Real-time Support in Fault-tolerant CORBA," IEEE Workshop on Dependable Middleware-Based Systems, Washington, D.C., June 23-26, 2002.

W32 Chris Gill, Joe Loyall, Rick Schantz, and Douglas C. Schmidt, "Lessons Learned From Using Adaptive DOC Middleware in Real Application Contexts," IEEE Workshop on Dependable Middleware-Based Systems, Washington, D.C., June 23-26, 2002.

W31 M. Mousavi, G. Russello, M. Chaudron, M. Reniers, T. Basten, A. Corsaro, S. Shukla, R. Gupta, and D.C. Schmidt, "Aspects + GAMMA = AspectGAMMA A Formal Framework for Aspect-Oriented Specification," Proceedings of Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design Workshop, Enschede, The Netherlands, April 2002.

W30 Joseph K. Cross and Douglas C. Schmidt, "Meta-Programming Techniques for Distributed Real-time and Embedded Systems," Proceedings of the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA, January, 2002.

W29 Douglas C. Schmidt and Mayur Deshpande and Carlos O'Ryan, "Operating System Performance in Support of Real-time Middleware," Proceedings of the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA, January, 2002.

W28 Christopher D. Gill, Ron Cytron, and Douglas C. Schmidt, "Middleware Scheduling Optimization Techniques for Distributed Real-Time and Embedded Systems," Proceedings of the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA, January, 2002.

W27 Douglas C. Schmidt, "Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems," EMSOFT 2001: First Workshop on Embedded Software, Lake Tahoe, California, October, 8th-10th, 2001.

W26 Darrell Brunsch, Carlos O'Ryan, and Douglas C. Schmidt, "Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA," Proceedings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.

W25 Irfan Pyarali, Marina Spivak, Douglas C. Schmidt, and Ron Cytron, "Optimizing Thread-Pool Strategies for Real-Time CORBA," Proceedings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.

W24 Yamuna Krishnamurthy, Vishal Kachroo, David A. Karr, Craig Rodrigues, Joseph P. Loyall, Richard Schantz, and Douglas C. Schmidt, "Integration of QoS-enabled Distributed Object Computing Middleware for Developing Next-generation Distributed Applications," Proceed-

ings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.

W23 Ossama Othman and Douglas C. Schmidt, Optimizing Distributed system Performance via Adaptive Middleware Load Balancing, Proceedings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.

W22 Pradeep Gore, Douglas C. Schmidt, Carlos O'Ryan, and Ron Cytron, "Designing and Optimizing a Scalable CORBA Notification Service," Proceedings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.

W21 Douglas C. Schmidt and Adam Porter, "Leveraging Open-Source Processes to Improve the Quality and Performance of Open-Source Software," Proceedings of the 1st Workshop on Open Source Software Engineering, ICSE 23, Toronto, Canada, May 15, 2001.

W20 Christopher D. Gill, David Levine, Douglas C. Schmidt, "Towards Real-Time Adaptive QoS Management in Middleware for Embedded Computing Systems," Fourth Annual Workshop on High Performance Embedded Computing, MIT Lincoln Laboratory, September 20-22, 2000.

W19 Christopher D. Gill, Fred Kuhns, David Levine, Douglas C. Schmidt, Bryan S. Doerr, and Richard E. Schantz, "Applying Adaptive Real-time Middleware to Address Grand Challenges of COTS-based Mission-Critical Real-Time Systems," Proceedings of the 1st International Workshop on Real-Time Mission-Critical Systems: Grand Challenge Problems, IEEE, Phoenix, Arizona, November 30, 1999.

W18 Carlos O'Ryan, Douglas C. Schmidt, David Levine, and Russell Noseworthy, "Applying a Real-time CORBA Event Service to Large-scale Distributed Interactive Simulation," 5th International Workshop on Object-oriented Real-Time Dependable Systems, IEEE, Monterey, CA, November 15-18, 1999.

W17 Fred Kuhns, Carlos O'Ryan, Douglas C. Schmidt, and Jeff Parsons, "The Design and Performance of a Pluggable Protocols Framework for Object Request Broker Middleware," Proceedings of the IFIP Sixth International Workshop on Protocols For High-Speed Networks (PfHSN '99), Salem, MA, August 25-27, 1999.

W16 David Levine, Sergio Flores-Gaitan, and Douglas C. Schmidt, "Measuring OS Support for Real-time CORBA ORBs," Proceedings of the Fourth International IEEE Workshop on Object-oriented Real-time Dependable Systems (WORDS'99), Santa Barbara, California, January 27-29, 1999.

W15 Douglas C. Schmidt, Rajeev Bector, David Levine Sumedh Mungee, and Guru Parulkar, "TAO: a Middleware Framework for Real-time ORB Endsystems," Proceedings of the Workshop on Middleware for Real-Time Systems and Services, held in conjunction with IEEE Real-time Systems Symposium, San Francisco, CA, December 2nd, 1997.

W14 Aniruddha Gokhale, Tim Harrison, Douglas C. Schmidt, and Guru Parulkar, "Operating System Support for Real-time CORBA," *Proceedings of the 5th International Workshop on Object-Orientation in Operating Systems*: IWOOOS 1996 workshop, October 27-28, 1996, Seattle, Washington.

W13 Douglas C. Schmidt, Guru Parulkar, and Chuck Cranor, "Gigabit CORBA - High-Performance Distributed Object Computing," Gigabit Networking Workshop (GBN'96), 24 March 1996, San Francisco, in conjunction with INFOCOM '96.

W12 Douglas C. Schmidt, "Acceptor and Connector: Design Patterns for Actively and Passively Initializing Network Services." Workshop on Pattern Languages of Object-Oriented Programs at ECOOP '95, August 7–1, 1995, Aarhus, Denmark.

W11 Douglas C. Schmidt, "High-Performance Event Filtering for Dynamic Multi-point Applications," Proceedings of the 1st Workshop on High Performance Protocol Architectures (HIPPARCH), INRIA, Sophia Antipolis, France, December, 1994, p 1-8.

W10 Douglas C. Schmidt, "Flexible Configuration of High-Performance Object-Oriented Distributed Communication Systems," 9th OOPSLA Conference, invited paper to the Workshop on Flexibility in Systems Software, ACM, Portland, Oregon, October, 1994, pp. 1-4.

W9 Douglas C. Schmidt and Tatsuya Suda, "Measuring the Impact of Alternative Parallel Process Architectures on Communication Subsystem Performance," *Proceedings of the Proceedings*

of the 4th International Workshop on Protocols for High-Speed Networks, IFIP, Vancouver, British Columbia, August, 1994, pp. 103-118.

W8 Douglas C. Schmidt and Tatsuya Suda, "The Service Configurator Framework: An Extensible Architecture for Dynamically Configuring Concurrent, Multi-service Network Daemons," *Proceedings of the 2nd International Workshop on Configurable Distributed Systems*, IEEE, Pittsburgh, PA, March 21-23, 1994, pp. 190-201.

W7 Douglas C. Schmidt, Burkhard Stiller, Tatsuya Suda, and Martina Zitterbart, "Tools for Generating Application-Tailored Multimedia Protocols on Heterogeneous Multi-Processor Platforms," *Proceedings of the 2nd Workshop on High-Performance Communications Subsystems*, IEEE, Williamsburg, Virginia, September 1-3, 1993, pp. 1-7.

W6 Douglas C. Schmidt and Tatsuya Suda, "A Framework for Developing and Experimenting with Parallel Process Architectures to Support High-Performance Transport Systems," *Proceedings of the 2nd Workshop on High-Performance Communications Subsystems*, IEEE, Williamsburg, Virginia, September 1-3, 1993, pp. 1-8.

W5 Tatsuya Suda, Douglas C. Schmidt, Donald F. Box, Duke Hong and Hung Huang, "High Speed Networks," *Proceedings of the International Computer World Symposium '92*, Kobe, Japan, November, 1992.

W4 Hung K. Huang, Douglas C. Schmidt, Donald F. Box, Kazu Shimono, Girish Kotmire, Unmesh Rathi, and Tatsuya Suda, "ADAPTIVE: A Prototyping Environment for Transport Systems." *Proceedings of the 4th International Workshop on Computer Aided Modeling, Analysis, and Design of Communication Links and Networks (CAMAD)*, IEEE, Montreal, Canada, September, 1992.

W3 Donald F. Box, Douglas C. Schmidt, and Tatsuya Suda, "Alternative Approaches to ATM/-Internet Interoperation," *Proceedings of the 1st Workshop on the Architecture and Implementation of High-Performance Communication Subsystems*, IEEE, Tucson, Arizona, February 17-19, 1992, pp. 1-5.

W2 Douglas C. Schmidt and Richard Selby "Modeling Software Interconnectivity," *Proceedings of the 22nd Symposium on the Interface: Computer Science and Statistics*, East Lansing, MI, May, 1990.

W1 Richard W. Selby, Greg James, Kent Madsen, Joan Mahoney, Adam A. Porter, and Douglas C. Schmidt "Classification Tree Analysis Using the Amadeus Measurement and Empirical Analysis System," *Proceedings of the 14th Annual Software Engineering Workshop at NASA Software Engineering Laboratory*, College Park, Maryland, November, 1989, pp. 239-250.

- **Trade Magazine and Newsletter Publications**

M74 Douglas C. Schmidt and Jules White, "Why Don't Big Companies Keep Their Computer Systems Up-to-date?," *The Conversation*, September 26, 2017.

M73 Douglas C. Schmidt, Accelerating the Industrial Internet with the OMG Data Distribution Service, Real-time Innovations, January 2014.

M72 Douglas C. Schmidt and Ron Guida, "Elastic Application Platforms for Cloud Computing," HPC In the Cloud, September 2010.

M71 Douglas C. Schmidt and Ron Guida, "Achieving Ultra High Performance in the Cloud," HPC In the Cloud, August 2010.

M70 Douglas C. Schmidt, "Building Ultra High Performance Computing Applications with Zircon Software," Programmer's Paradise, May 31st, 2010.

M69 Egon Wuchner, Andrey Nechypurenko, Jules White, and Douglas C. Schmidt, "Das Generic Eclipse Modeling System (GEMS): Skalierbare Domanenmodellierung Leicht(er) Gemacht," SIGS ObjectSpektrum, June, 2007.

M68 Jules White, Douglas C. Schmidt, Andrey Nechypurenko, and Egon Wuchner, Introduction to the Generic Eclipse Modeling System, Eclipse Magazine, Volume 06, January, 2007.

M67 Bala Natarajan, Douglas C. Schmidt, and Steve Vinoski, "Object Interconnections: The CORBA Component Model Part 4: Implementing Components with CCM, C/C++ Users Journal, October, 2004.

M66 Bala Natarajan, Douglas C. Schmidt, and Steve Vinoski, "The CORBA Component Model Part 3: The CCM Container Architecture and Component Implementation Framework," C/C++ Users Journal, September, 2004.

M65 Douglas C. Schmidt, Richard Schantz, Aniruddha Gokhale, and Joe Loyall, "Middleware R&D Challenges for Distributed Real-time and Embedded Systems," SIGBED Review, Volume 1, No. 1, April 2004.

M64 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: The CORBA Component Model: Part 2: Defining Components with the IDL 3.x Types," C/C++ Users Journal, May, 2004.

M63 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: The CORBA Component Model: Part 1: Evolving Towards Components," C/C++ Users Journal, February, 2004.

M62 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: XML Reflection for CORBA," C/C++ Users Journal, December, 2003.

M61 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: CORBA Metaprogramming Mechanisms, Part 1: Portable Interceptors Concepts and Components," C/C++ Users Journal, March, 2003.

M60 Douglas C. Schmidt and Steve Vinoski, Object Interconnections: "Dynamic CORBA, Part 4: The Interface Repository," C/C++ Users Journal, January, 2003.

M59 Douglas C. Schmidt, "Voice of the Customer: An Interview," Raytheon Technology Today, spring 2003, Volume 2, Issue 1.

M58 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: Dynamic CORBA, Part 3: The Dynamic Skeleton Interface," C/C++ Users Journal, November, 2002.

M57 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: Dynamic CORBA, Part 2: Dynamic Any," C/C++ Users Journal, September, 2002.

M56 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: Dynamic CORBA, Part 1: The Dynamic Invocation Interface," C/C++ Users Journal, July, 2002.

M55 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: Real-time CORBA, Part 4: Protocol Selection and Explicit Binding," C/C++ Users Journal, May, 2002.

M54 Douglas C. Schmidt and Steve Huston, "Why Standards Alone Won't Get You Portable Software And How to Make Open Source Development Work for You," InformIT: Focus on C++, Addison-Wesley.

M53 Douglas C. Schmidt and Steve Vinoski, "Real-time CORBA Part 3: Thread Pools and Synchronizers," C/C++ Users Journal, March, 2002.

M52 Douglas C. Schmidt and Steve Vinoski, "Real-time CORBA, Part 2: Applications and Priorities," C/C++ Users Journal, January, 2002.

M51 Douglas C. Schmidt and Steve Vinoski, "Real-time CORBA, Part 1: Motivation and Overview," *C/C++ Users Journal*, October, 2001.

M50 Douglas C. Schmidt and Steve Vinoski, "CORBA and XML, Part 3: SOAP and Web Services," *C/C++ Users Journal*, September, 2001.

M48 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: CORBA and XML, Part 2: XML as CORBA Data," *C/C++ Users Journal*, July, 2001.

M47 Douglas C. Schmidt and Steve Vinoski, "Object Interconnections: CORBA and XML, Part 1: Versioning," *C/C++ Users Journal*, May, 2001.

M46 Douglas C. Schmidt and Steve Vinoski, "Standard C++ and the OMG C++ Mapping: Server-side Mappings and Pseudo-Objects," *C/C++ Users Journal*, April, 2001.

M45 Douglas C. Schmidt and Steve Vinoski, "Standard C++ and the OMG C++ Mapping," *C/C++ Users Journal*, January, 2001.

M44 Douglas C. Schmidt and Steve Vinoski, "The History of the OMG C++ Mapping", *C/C++ Users Journal*, November, 2000.

M43 Douglas C. Schmidt and Steve Vinoski, "An Overview of the OMG CORBA Messaging Quality of Service (QoS) Framework," *C++ Report*, SIGS, Vol. 12, No 3, March, 2000.

M42 Douglas C. Schmidt, "Monitor Object – an Object Behavior Pattern for Concurrent Programming," *C++ Report*, SIGS, Vol. 12., No. 4. May 2000.

M41 Alexander B. Arulanthu, Carlos O'Ryan, Douglas C. Schmidt, and Michael Kircher, "Applying Patterns and Components to Develop an IDL Compiler for CORBA AMI Callbacks," *C++ Report*, SIGS, Vol. 12, No. 3, March, 2000.

M40 David Levine, Chris Gill, and Douglas C. Schmidt, "Object Lifetime Manager – A Complementary Pattern for Controlling Object Creation and Destruction," *C++ Report*, SIGS, Vol. 11, No. 11, November/December, 1999.

M39 Douglas C. Schmidt, Steve Vinoski, and Nanbor Wang, "Collocation Optimizations for CORBA," *C++ Report*, SIGS, Vol. 11, No. 10, October, 1999.

M38 Douglas C. Schmidt, "Strategized Locking, Thread-safe Decorator, and Scoped Locking: Patterns and Idioms for Simplifying Multi-threaded C++ Components," *C++ Report*, SIGS, Vol. 11, No. 9, September, 1999.

M37 Douglas C. Schmidt and Steve Vinoski, "Time-Independent Invocation and Interoperable Routing," *C++ Report*, SIGS, Vol. 11, No 5, May, 1999.

M36 Michael Kircher and Douglas C. Schmidt, "Dove: A Distributed Object Visualization Environment," *C++ Report*, SIGS, Vol. 11, No 3, March, 1999.

M35 Douglas C. Schmidt, "Wrapper Facade: A Structural Pattern for Encapsulating Functions within Classes," *C++ Report*, SIGS, Vol. 11, No 2, February, 1999.

M34 Douglas C. Schmidt and Steve Vinoski, "Programming Asynchronous Method Invocation with CORBA Messaging," *C++ Report*, SIGS, Vol. 11, No 2, February, 1999.

M33 Douglas C. Schmidt "Why Software Reuse has Failed and How to Make It Work for You," C++ Report, SIGS, Vol. 11, No. 1, January, 1999.

M32 Douglas C. Schmidt, "An Architectural Overview of the ACE Framework: A Case-study of Successful Cross-platform Systems Software Reuse," USENIX login magazine, Tools special issue, November, 1998.

M31 Douglas C. Schmidt, "GPERF: A Perfect Hash Function Generator," C++ Report, SIGS, Vol. 10, No. 10, November/December, 1998.

M30 Douglas C. Schmidt and Steve Vinoski, "Introduction to CORBA Messaging," SIGS, Vol. 10, No 10, November/December, 1998.

M29 Douglas C. Schmidt and Steve Vinoski, "C++ Servant Managers for the Portable Object Adapter," SIGS, Vol. 10, No 8, September, 1998.

M28 Chris Cleeland and Douglas C. Schmidt, "External Polymorphism, An Object Structural Pattern for Transparently Extending C++ Concrete Data Types," C++ Report, SIGS, Vol. 10, No. 6, July/August, 1998.

M27 Douglas C. Schmidt and Irfan Pyarali, "Strategies for Implementing POSIX Condition Variables on Win32," C++ Report, SIGS, Vol. 10, No. 5, June, 1998.

M26 Douglas C. Schmidt and Steve Vinoski, "Using the Portable Object Adapter for Transient and Persistent CORBA Objects," C++ Report, SIGS, Vol. 10, No 4. May, 1998.

M25 Douglas C. Schmidt, "Applying Design Patterns to Simplify Signal Handling," C++ Report, SIGS, Vol. 10, No. 4, May, 1998.

M24 Douglas C. Schmidt, Tim H. Harrison, and Nat Pryce, "Thread-specific Storage: an Object Behavioral Pattern for Efficiently Accessing per-Thread State," C++ Report, SIGS, Vol. 9, No. 10, November/December, 1997

M23 Douglas C. Schmidt and Steve Vinoski, "Object Adapters: Concepts and Terminology," C++ Report, SIGS, Vol. 9, No 11. November/December, 1997.

M22 Prashant Jain and Douglas C. Schmidt, "Dynamically Configuring Communication Services with the Service Configurator Pattern," *C++ Report*, SIGS, Vol. 9, No. 6, June, 1997.

M21 Douglas C. Schmidt and Steve Vinoski, "Overcoming Drawbacks in the OMG Events Service," *C++ Report*, SIGS, Vol. 9, No 6. June, 1997.

M20 Douglas C. Schmidt and Steve Vinoski, "OMG Event Object Service," *C++ Report*, SIGS, Vol. 9, No 2. February, 1997.

M19 Prashant Jain and Douglas C. Schmidt, "Experiences Converting a C++ Communication Framework to Java," *C++ Report*, SIGS, Vol. 9, No. 1, January, 1997.

M18 Douglas C. Schmidt, "Lessons Learned Building Reusable OO Telecommunication Software," Multiuse Express, Lucent Technologies, Vol. 4, No. 6, December, 1996.

M17 Douglas C. Schmidt and Steve Vinoski, "Distributed Callbacks and Decoupled Communication in CORBA," *C++ Report*, SIGS, Vol. 8, No 9. October, 1996.

M16 Timothy H. Harrison and Douglas C. Schmidt, "Evaluating the Performance of OO Network Programming Toolkits," *C++ Report*, SIGS, Vol. 8, No 7. July/August 1996.

M15 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Programming Techniques for Multi-threaded Servers – the Thread-per-Session Concurrency Model," *C++ Report*, SIGS, Vol. 8, No 7. July/August 1996.

M14 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Programming Techniques for Multi-threaded Servers – the Thread-Pool Concurrency Model," *C++ Report*, SIGS, Vol. 8, No 4. April 1996.

M13 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Programming Techniques for Multi-threaded Servers – the Thread-per-Request Concurrency Model," *C++ Report*, SIGS, Vol. 8, No 2. February 1996.

M12 Douglas C. Schmidt, "A Design Pattern for Actively Initializing Network Services," *C++ Report*, SIGS, Vol. 8, No. 1, January 1996.

M11 Douglas C. Schmidt, "Design Patterns for Initializing Network Services: Introducing the Acceptor and Connector Patterns," *C++ Report*, SIGS, Vol. 7, No. 9, November/December 1995.

M10 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Server-side Distributed Programming Techniques," Object Interconnections Column, *C++ Report*, SIGS, Vol. 7, No. 8, October 1995.

M9 Douglas C. Schmidt and Steve Vinoski, "Comparing Alternative Client-side Distributed Programming Techniques," Object Interconnections Column, *C++ Report*, SIGS, Vol. 7, No. 4, May 1995.

M8 Douglas C. Schmidt and Paul Stephenson, "Using Design Patterns to Evolve System Software from UNIX to Windows NT," *C++ Report*, SIGS, Vol. 7, No. 3, March/April 1995, pp. 27-39.

M8 Douglas C. Schmidt and Steve Vinoski, "Distributed Object Computing by Example," Object Interconnections Column, *C++ Report*, SIGS, Vol. 7, No. 2, February 1995.

M7 Douglas C. Schmidt and Steve Vinoski, "Distributed Object Computing with C++," Object Interconnections Column, *C++ Report*, SIGS, Vol. 7, No. 1, January 1995.

M6 Douglas C. Schmidt, "Transparently Parameterizing Synchronization Mechanisms into a Concurrent Distributed Application," *C++ Report*, SIGS, Vol. 6, No. 5, July/August 1994, pp. 1-10.

M5 Douglas C. Schmidt, "A Domain Analysis of Network Daemon Design Dimensions," *C++ Report*, SIGS, Vol. 6, No. 3, March/April, 1994, pp. 1-12.

M4 Douglas C. Schmidt, "The Object-Oriented Design and Implementation of the Reactor: A C++ Wrapper for UNIX I/O Multiplexing," *C++ Report*, SIGS, Vol. 5, No. 7, September, 1993, pp. 1-14.

M3 Douglas C. Schmidt, "The Reactor: An Object-Oriented Interface for Event-Driven UNIX I/O Multiplexing," *C++ Report*, SIGS, Vol. 5, No. 2, February, 1993, pp. 1-12.

M2 Douglas C. Schmidt, "IPC_SAP: An Object-Oriented Interface to Operating System Interprocess Communication Services," *C++ Report*, SIGS, Vol. 4, No. 8, November/December, 1992, pp. 1-10.

M1 Douglas C. Schmidt, "Systems Programming with C++ Wrappers: Encapsulating Interprocess Communication Services with Object-Oriented Interfaces," *C++ Report*, SIGS, Vol. 4, No. 7, September/October, 1992, pp 1-6.

- **Refereed Short Papers, Posters, and Demos**

P14 Maria Powell, Marcelino Rodriguez Cancio, David Young, William Nock, Beshoy Abdelmessih, Amy Zeller, Irvin Perez Morales, Peng Zhang, C. Gaelyn Garrett, Douglas Schmidt, Jules White, and Alexander Gelbard, "Decoding Phonation with Artifical Intelligence: Proof of Concept," Poster Proceedings in the 13th International Conference on Advances in Quantitative Laryngology, Voice and Speech Research, Montreal, Canada, June 2-4th, 2019.

P13 Akram Hakiri, Berthou Pascal, Gayraud Thierry, Aniruddha Gokhale, Joe Hoffert, and Douglas C. Schmidt, "SIP-based QoS Support and Session Management for DDS-based Distributed Real-time and Embedded Systems," Poster Proceedings of the 5th ACM International Conference on Distributed Event-based Systems (DEBS'11), New York City, NY, USA, July 11-15, 2011.

P12 Jules White, Andrey Nechypurenko, Egon Wuchner, and Douglas C. Schmidt "Automatic Role-based Constraint Solving for Real-Time and Embedded Systems: An Approach to Modeling Guidance", poster paper at the 14th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS), March 26th-29th, 2007, Tucson, Arizona.

P11 John Kinnebrew, Nishanth Shankaran, Gautam Biswas, and Douglas Schmidt, "A Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-Time Applications," Poster paper at the Twenty-First National Conference on Artificial Intelligence, July 16 20, 2006, Boston, Massachusetts.

P10 Jai Balasubramanian, Nishanth Shankar, Douglas C. Schmidt, Gautam Biswas, Patrick Lardieri, Ed Mulholland, and Tom Damiano, "A Framework for (Re)Deploying Components in Distributed Realtime and Embedded Systems," poster paper at the Dependable and Adaptive Distributed Systems, Track of the 21st ACM Symposium on Applied Computing, April 23 -27, 2006, Bourgogne University, Dijon, France.

P9 Jules White and Douglas C. Schmidt, Reducing Enterprise Product Line Architecture Deployment Costs via Model-Driven Deployment and Configuration Testing, Poster paper at the 13th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS '06) March 27th-30th, 2006, University of Potsdam, Potsdam, Germany.

P8 Arvind S. Krishna, Aniruddha Gokhale, Douglas C. Schmidt, John Hatcliff, and Venkatesh Prasad Ranganat, "Towards Highly Optimized Real-time Middleware for Software Product-line Architectures," Proceedings of the Work-In-Progress session at the 26th IEEE Real-Time Systems Symposium, December 5-8, 2005, Miami, Florida.

P7 Gan Deng, Douglas Schmidt, and Aniruddha Gokhale, "Supporting Configuration and Deployment of Component-based DRE Systems Using Frameworks, Models, and Aspects," Poster Session of the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programing, Systems, Languages and Applications (OOPSLA 2005), San Diego, CA, October 16-20, 2005.

P6 Jules White, Douglas Schmidt, and Aniruddha Gokhale, "The J3 Process for Building Autonomic Enterprise Java Bean Systems," Proceedings of the International Conference on Autonomic Computing (ICAC 2005), Seattle, WA, June 2005 (short paper).

P5 Aniruddha Gokhale, Arvind Krishna and Douglas C. Schmidt, "CoSMIC: Addressing Crosscutting Deployment and Configuration Concerns of Distributed Real-time and Embedded Systems via Aspect-oriented and Model-driven Software Development," Demo Session, Fourth International Conference on Aspect-oriented Software Development (AOSD), Chicago, IL, Mar 2005.

P4 Aniruddha Gokhale, Krishnakumar Balasubramanian, Jaiganesh Balasubramanian, Arvind Krishna, George Edwards, Gan Deng, Jeff Parsons, Tao Lu, Balachandran Natarajan and Douglas C. Schmidt, "CoSMIC: Addressing Crosscutting Deployment and Configuration Concerns in QoS-sensitive Distributed Systems," 19th ACM OOPSLA Conference, Poster Session, Vancouver, Canada, Oct 2004.

P3 Joseph K. Cross and Douglas C. Schmidt, "Meta-Programming Techniques to Declaratively Optimize Middleware Policies and Mechanisms," Poster session at the IFIP/ACM Middleware 2001 International Conference on Distributed Systems Platforms, Heidelberg, Germany, November 12-16, 2001.

P2 Aniruddha Gokhale and Douglas C. Schmidt, "Design Principles and Optimizations for High Performance ORBs," ACM, *OOPSLA 97*, Poster Session, Oct 1997, Atlanta, GA, USA.

P1 Douglas C. Schmidt, "Performance Experiments on Alternative Methods for Structuring Active Objects in High-Performance Parallel Communication Systems," 9th ACM OOPSLA Conference, poster paper, Portland, Oregon, October, 1994, pp. 1-12.

- **Editorials and Book Forewords**

E109 Douglas C. Schmidt and Nicholas Guertin, "Toward Technical Reference Frameworks to Support Large-Scale Systems of Systems," SEI Blog, December 20th, 2021.

E108 Thomas Evans, Michael Gagliardi, Joseph Kostial, Nicholas Reimer, and Douglas C. Schmidt, "Technical Issues in Navigating the Transition from Sustainment to Engineering Software-Reliant Systems," SEI Blog, December 6th, 2021.

E107 Douglas C. Schmidt, "The Latest Work from the SEI: Coordinated Vulnerability Disclosure, Cybersecurity Research, Cyber Risk and Resilience, and the Importance of Fostering Diversity in Software Engineering," SEI Blog, September 6th, 2021.

E106 Thomas Evans, Michael Gagliardi, Joseph Kostial, Nicholas Reimer, and Douglas C. Schmidt, "Navigating People Concerns when Transitioning from Sustainment to Engineering Software-Reliant Systems," SEI Blog, August 18th, 2021.

E105 Anita Carleton, John Robert, Mark Klein, Doug Schmidt, Forrest Shull, John Foreman, Ipek Ozkaya, Robert Cunningham, Charlie Holland, Erin Harper, and Edward Desautels, "Architecting the Future of Software Engineering: A Research and Development Roadmap", SEI Blog, July 12, 2021.

E104 Douglas C. Schmidt, "The Latest Work from the SEI: Artificial Intelligence, DevSecOps, and Security Incident Response," SEI Blog, July 5th, 2021.

E103 Tom Evans, Mike Gagliardi, Joe Kostial, Nicholas Reimer, and Douglas C. Schmidt, "Process Concerns When Navigating the Transition from Sustainment to Engineering Software-Reliant Systems," SEI Blog, June 14th, 2021.

E102 Douglas C. Schmidt, "The Latest Work from the SEI: Privacy, Ransomware, Digital Engineering, and the Solar Winds Hack," SEI Blog, April 5th, 2021.

E101 Douglas C. Schmidt, "Top 10 Blog Posts of 2020," SEI Blog, January 11th, 2021.

E100 Tom Evans, Mike Gagliardi, Joe Kostial, Nicholas Reimer, and Douglas C. Schmidt, "Shifting from Software Sustainment to Software Engineering in the DoD," SEI Blog, November 30th, 2020.

E99 Douglas C. Schmidt, "The Latest Work from the SEI: Microservices, Ransomware, and Agile in Government," SEI Blog, September 7th, 2020.

E98 Douglas C. Schmidt, "The Latest Work from the SEI: DevSecOps, Artificial Intelligence, and Cybersecurity Maturity Model Certification," SEI Blog, May 25th, 2020.

E97 Douglas C. Schmidt, "The Latest Work from the SEI: Penetration Testing, Artificial Intelligence, and Incident Management," SEI Blog, January 20th, 2020.

E96 Douglas C. Schmidt, "Top 10 Blog Posts of 2019," SEI Blog, January 6th, 2020.

E95 Douglas C. Schmidt, "The Latest Work from the SEI: Microservices, Ransomware, and Agile in Government", SEI Blog, September 7th, 2019.

E94 Douglas C. Schmidt, "The Latest Work from the SEI: DevSecOps, Artificial Intelligence, and Cybersecurity Maturity Model Certification", SEI Blog, May 25th, 2020.

E93 Douglas C. Schmidt, "The Latest Work from the SEI: Penetration Testing, Artificial Intelligence, and Incident Management", SEI Blog, January 20th, 2019.

E92 Douglas C. Schmidt, "The Top 10 Blog Posts of 2019, SEI Blog, January 6th, 2020.

E91 Nick Guertin, Douglas C. Schmidt, and William Scherlis, "Impacts and Recommendations for Achieving Modular Open Systems Architectures, SEI Blog, September 23rd, 2019.

E90 Douglas C. Schmidt, "The Latest Work from the SEI: AI, Deepfakes, Automated Alert Handling, and Cyber Intelligence," SEI Blog, September 2nd, 2019.

E89 Douglas C. Schmidt, "The Latest Research from the SEI in DevSecOps, Threat Modeling, and Insider Threat," SEI Blog, May 28th, 2019.

E88 Douglas C. Schmidt, "Deep Learning, Agile-DevOps, and Cloud Security: The Top 10 Blog Posts of 2018," SEI Blog, January 7th, 2019.

E87 Nick Guertin, Douglas C. Schmidt, and William Scherlis, "The Technical Architecture for Product Line Acquisition in the DoD," SEI Blog, May 6th, 2019.

E86 Nick Guertin, Douglas C. Schmidt, and William Scherlis, "The Organizational Impact of a Modular Product Line Architecture in DoD Acquisition," SEI Blog, April 29th, 2019.

E85 Nick Guertin, Douglas C. Schmidt, and William Scherlis, "Towards a New Model of Acquisition: Product-Line Architectures for the DoD," SEI Blog, March 11th, 2019.

E84 Douglas C. Schmidt, "Deep Learning, Agile-DevOps, and Cloud Security: The Top 10 Blog Posts of 2018," SEI Blog, January 7th, 2019.

E83 Nick Guertin, Douglas C. Schmidt, and William Scherlis, "Emerging Opportunities in Modularity and Open Systems Architectures," SEI Blog, October 15th, 2018.

E82 Douglas C. Schmidt, "Learning, Cyber Intelligence, Managing Privacy and Security, and Network Traffic Analysis: The Latest Work from the SEI," SEI Blog, July 2, 2018.

E81 Douglas C. Schmidt, "Virtual Integration, Blockchain Programming, and Agile/DevOps: The Latest Work from the SEI," SEI Blog, May 28, 2018.

E80 Douglas C. Schmidt, "Fighting Chance: Arming the Analyst in the Age of Big Data," SEI Blog, March 26, 2018.

E79 Douglas C. Schmidt, "Agile/DevOps, Best Practices in Insider Threat, and Dynamic Design Analysis: The Latest Work from the SEI," SEI Blog, February 26, 2018.

E78 Douglas C. Schmidt, "Bitcoin, Blockchain, Machine Learning, and Ransomware: The Top 10 Posts of 2017," SEI Blog, January 8, 2018.

E77 Douglas C. Schmidt, "Cyber Warfare, Technical Debt, Network Border Protection, and Insider Threat: The Latest Work from the SEI," SEI Blog, November 27, 2017.

E76 Douglas C. Schmidt, "Coordinated Vulnerability Disclosure, Ransomware, Scaling Agile, and Android App Analysis: The Latest Work from the SEI,SEI Blog, " September 5, 2017.

E75 Douglas C. Schmidt, "Top 10 SEI Blog Posts of 201I," SEI Blog, July 10, 2017.

E74 Douglas C. Schmidt, "Supply Chain Risk Management, Network Situational Awareness, Software Architecture, and Network Time Protocol: The Latest Work from the SEI," SEI Blog, July 3, 2017.

E73 Douglas C. Schmidt, "Software Assurance, Data Governance, and Malware Analysis: The Latest Work from the SEI," SEI Blog, April 10, 2017.

E72 Douglas C. Schmidt, "Preventing DDoS Attacks, Scaling Agile, Insider Threat, and Software Architecture: The Latest Work from the SEI," SEI Blog, January 30th, 2017.

E71 Douglas C. Schmidt, "Autonomy, Robotics, Verification, DDoS Attacks, and Software Testing: The Top 10 Posts of 2016," SEI Blog, December 19th, 2016.

E70 Douglas C. Schmidt, "Cybersecurity Engineering, Performance, Risk, and Secure Coding: The Latest Work from the SEI," SEI Blog, November 28th, 2016.

E69 Douglas C. Schmidt, "Resilience, Secure Coding, Data Science, Insider Threat, and Scheduling: The Latest Research from the SEI," SEI Blog, October 17th, 2016.

E68 Douglas C. Schmidt, "Data Science, Blacklists, and Mixed-Critical Software: The Latest Research from the SEI," SEI Blog, September 5th, 2016.

E67 Douglas C. Schmidt and Carol Sledge, "A Naval Perspective on Open Systems Architecture," SEI Blog, July 11th, 2016.

E66 Douglas C. Schmidt, "Top 10 SEI Blog Posts of 2016," SEI Blog, July 4th, 2016.

E65 Douglas C. Schmidt, "Top 10 SEI Blog Posts of 2016, SEI Blog, July 4th, 2016. Douglas C. Schmidt, Situational Analysis, Software Architecture, Insider Threat, Threat Modeling, and Honeynets: The Latest Research from the SEI," SEI Blog, May 30th, 2016.

E64 Douglas C. Schmidt, "Threat Analysis Mapping, Connected Vehicles, Emerging Technologies, and Cyber-Foraging: The Latest Research from the SEI," SEI Blog, May 2nd, 2016.

E63 Douglas C. Schmidt, "The Top 10 Blog Posts of 2015: Technical Debt, DevOps, Graph Analytics, Secure Coding, and Testing," January 4th, 2016.

E62 Carol Sledge and Douglas C. Schmidt, "A Discussion on Open-Systems Architecture," SEI Blog November 23rd, 2015.

E61 Douglas C. Schmidt, "Agile, Architecture Fault Analysis, the BIS Wassenaar Rule, and Computer Network Design," SEI Blog, September 7, 2015.

E60 Douglas C. Schmidt, "Testing, Agile Metrics, Fuzzy Hashing, Android, and Big Data" SEI Blog, July 13, 2015.

E59 Douglas C. Schmidt, "Resilience, Model-Driven Engineering, Software Quality, and Android App Analysis," SEI Blog, May 18, 2015.

E58 Douglas C. Schmidt, "Resilience, Metrics, Sustainment, and Software Assurance," SEI Blog, February 23, 2015.

E57 Douglas C. Schmidt, "Software Assurance, Social Networking Tools, Insider Threat, and Risk Analysis," SEI Blog, January 19th 2015.

E56 Douglas C. Schmidt, "The 2014 Year in Review," SEI blog, December 22nd, 2014.

E55 Douglas C. Schmidt, Android, Heartbleed, Testing, and DevOps: An SEI Blog Mid-Year Review, SEI blog, June 30th, 2014.

E54 Douglas C. Schmidt, “The Importance of Automated Testing in Open Systems Architecture Initiatives,” SEI blog, March 23rd, 2014.

E53 Douglas C. Schmidt, “How Vanderbilt’s Secret Software Lab Is Saving America,” gizmodo.com, January 10th, 2014.

E52 Douglas C. Schmidt, “The SEI Blog: the Research Year in Review,” SEI blog, December 23rd, 2013.

E51 Douglas C. Schmidt “The Architectural Evolution of DoD Combat Systems,” SEI blog, November 25th, 2013.

E50 Douglas C. Schmidt, “Three Qs: Vanderbilt Professor Douglas Schmidt,” GE’s Industrial Internet blog, September 10th, 2013.

E49 Douglas C. Schmidt, “Towards Affordable DoD Combat Systems in the Age of Sequestration,” SEI Blog, September 9th, 2013.

E48 Douglas C. Schmidt, “Ten Tech Terms Everyone Needs to Know for 2014,” Yahoo Tech News, August 12, 2013.

E47 Douglas C. Schmidt and Philippe Fauchet, “Students Must Stay to Better Workforce,” The Tennessean, August 6th, 2013.

E46 Douglas C. Schmidt, “Learning in MOOC Years,” Vanderbilt Magazine, Spring 2013.

E45 Douglas C. Schmidt, The SEI Blog: A Two-year Retrospective, SEI blog, April 1st, 2013.

E44 Douglas C. Schmidt, 2012: “The Research Year in Review,” SEI blog, December 24th, 2012.

E43 Douglas C. Schmidt, “Reflections on 20 Years of Architecture for Distributed Real-time and Embedded Systems by Douglas C. Schmidt,” SEI blog, October 29th, 2012.

E42 Douglas C. Schmidt, “Applying Agility to DoD Common Operating Platform Environment Initiatives,” SEI blog, July 30th, 2012.

E41 Douglas C. Schmidt, “Balancing Agility and Discipline at Scale,” SEI blog, July 23rd, 2012.

E40 Douglas C. Schmidt, “Strategic Management of Architectural Technical Debt,” SEI blog, July 16th, 2012.

E39 Douglas C. Schmidt, “Agile Methods: Tools, Techniques, and Practices for the DoD Community,” SEI blog, July 9th, 2012.

E38 Douglas C. Schmidt, “Applying Agile at-Scale for Mission-Critical Software-Reliant Systems,” SEI blog, July 2nd, 2012.

E37 Douglas C. Schmidt, “Toward Common Operating Platform Environments, Part 2: Understanding Success Drivers,” SEI blog, May 7th, 2012.

E36 Douglas C. Schmidt, “Toward Common Operating Platform Environments, Part 1: Doing More for Less,” SEI blog, April 30th, 2012.

E35 Douglas C. Schmidt, “The Road Ahead for SEI R&D in 2012,” SEI blog, December 26th, 2011.

E34 Douglas C. Schmidt, “A Summary of Key SEI R&D Accomplishments in 2011,” SEI blog, December 19th, 2011.

E33 Douglas C. Schmidt, “Bridging the Valley of Disappointment for DoD Software Research with SPRUCE,” SEI blog, November 7th, 2011.

E32 Douglas C. Schmidt, “The Growing Importance of Software Sustainment for the DoD, Part 2: SEI R&D Activities Related to Sustaining Software for the DoD”, SEI blog, August 15th, 2011.

E31 Douglas C. Schmidt, “The Growing Importance of Software Sustainment for the DoD, Part 1: Current Trends and Challenges”, SEI blog, August 1st, 2011.

E30 Douglas C. Schmidt, “New and Upcoming SEI Research Initiatives,” SEI blog, February 21st, 2011.

E29 Douglas C. Schmidt, “Advancing the Scope and Impact of SEI Research,” SEI blog, February 9th, 2011.

E28 Douglas C. Schmidt, Foreword to the book *Patterns xof Parallel Software Design*, by Jorge Luis Ortega Arjona, Wiley, 2010.

E27 Douglas C. Schmidt, Foreword to the book *Practical Software Factories in .NET* by Gunther Lenz and Christoph Wienands, Apress, 2006.

E26 Douglas C. Schmidt, Guest editorial of the IEEE Computer Special Issue on Model Driven Development, February 2006.

E25 Douglas C. Schmidt, Guest editorial for IEEE Networks magazine Special Issue on Middleware Technologies for Future Communication Networks, January 2004.

E24 Douglas C. Schmidt, Foreword to the book *Fundamentals of Distributed Object Systems: The CORBA Perspective*, by Zahir Tari and Omran Bukhres, Wiley and Sons, 2001.

E23 Douglas C. Schmidt, Foreword to the book *Design Patterns in Communication Software*, edited by Linda Rising and published by Cambridge University Press, 2000.

E22 Douglas C. Schmidt, "Trends in Distributed Object Computing" editorial for the special issue on Distributed Object-Oriented Systems appearing in the Parallel and Distributed Computing Practices journal, edited by Maria Cobb and Kevine Shaw, Vol. 3, No. 1, March 2000.

E21 Douglas C. Schmidt, "Object-Oriented Application Frameworks," guest editorial for the of the Communications of the ACM, Special Issue on Object-Oriented Application Frameworks, Vol. 40, No. 10, October 1997.

E20 Douglas C. Schmidt, "Recent Advances in Distributed Object Computing," guest editorial for the IEEE Communications Magazine feature topic issue on Distributed Object Computing, Vol. 14, No. 2, February, 1997.

E19 Douglas C. Schmidt, Guest editorial for the USENIX Computing Systems Special Issue on Distributed Object Computing Vol. 9, No. 4, November/December, 1996.

E18 Douglas C. Schmidt, "Software Patterns," guest editorial for Communications of the ACM, Special Issue on Patterns and Pattern Languages, Vol. 39, No. 10, October 1996.

E17 Douglas C. Schmidt, "Using Design Patterns to Develop Reuseable Object-Oriented Software," Strategic Directions in Computing Research OO Working Group conference, MIT, June 14-15, 1996.

E16 Douglas C. Schmidt, "The Last Waltz," C++ Report, SIGS, Vol. 11, No. 4, April 1999.

E15 Douglas C. Schmidt, "Patterns++ - the Next Generation," C++ Report, SIGS, Vol. 9, No. 4, April 1997.

E14 Douglas C. Schmidt, "CORBA: CASE for the late '90s?" C++ Report, SIGS, Vol. 9, No. 2, February 1997.

E13 Douglas C. Schmidt, "Java: Friend or Foe," C++ Report, SIGS, Vol. 9, No. 1, January 1997.

E12 Douglas C. Schmidt, "Promise Keepers," C++ Report, SIGS, Vol. 8, No. 11, November/December 1996.

E11 Douglas C. Schmidt, "The Programming Principle," C++ Report, SIGS, Vol. 8, No. 10, October 1996.

E10 Douglas C. Schmidt, "Pattern Forces," C++ Report, SIGS, Vol. 8, No. 9, September 1996.

E9 Douglas C. Schmidt, "The Secrets of Success for C++," C++ Report, SIGS, Vol. 8, No. 9, August 1996.

E8 Douglas C. Schmidt, "The C++ Decade," C++ Report, SIGS, Vol. 8, No. 9, August 1996.

E7 Douglas C. Schmidt, "Addressing the Challenge of Concurrent and Distributed Systems," C++ Report, SIGS, Vol. 8, No. 7, July 1996.

E6 Douglas C. Schmidt, "Delivering the Goods," C++ Report, SIGS, Vol. 8, No. 6, June 1996.

E5 Douglas C. Schmidt, "Problems with Process," C++ Report, SIGS, Vol. 8, No. 5, May 1996.

E4 Douglas C. Schmidt, "The Impact of Social Forces on Software Project Failures," C++ Report, SIGS, Vol. 8, No. 4, April 1996.

E3 Douglas C. Schmidt, "Reality Check," C++ Report, SIGS, Vol. 8, No. 3, March 1996.

E2 Douglas C. Schmidt, "Role Models for Success," C++ Report, SIGS, Vol. 8, No. 2, February 1996.

E1 Douglas C. Schmidt, "A Zest for Programming," C++ Report, SIGS, Vol. 8, No. 1, January 1996.

- **Technical Reports**

TR18 Douglas C. Schmidt, “Google Data Collection,” Vanderbilt University Technical Report #ISIS-20-201, August 15, 2018.

TR17 Gan Deng, Douglas C. Schmidt, Aniruddha Gokhale, “ Ensuring Deployment Predictability of Distributed Real-time and Embedded Systems,” Vanderbilt University Technical Report #ISIS-07-814, November 2007.

TR16 Jaiganesh Balasubramanian, Sumant Tambe, Chenyang Lu, Christopher Gill, Aniruddha Gokhale, and Douglas C. Schmidt, “FLARe: a Fault-tolerant Lightweight Adaptive Real-time Middleware for Distributed Real-time and Embedded, Systems,” Vanderbilt University Technical Report #ISIS-07-812, October 2007.

TR15 Shanshan Jiang, Yuan Xue, and Douglas Schmidt, “Minimum Disruption Service Composition and Recovery in Mobile Ad hoc Networks,” Vanderbilt University Technical Report #ISIS-06-711, December 2006.

TR14 Andrey Nechypurenko, Egon Wuchner, Jules White, Douglas C. Schmidt, “ Application of Aspect-based Modeling and Weaving for Complexity Reduction in the Development of Automotive Distributed Real-time Embedded Systems,” Vanderbilt University Technical Report #ISIS-06-709, July 2006.

TR13 James H. Hill, John M. Slaby, Steve Baker, Douglas C. Schmidt, “Predicting the Behavior for Components of the SLICE Scenario,” Vanderbilt University Technical Report #ISIS-05-608, October 2005.

TR12 Stoyan Paunov, James Hill, Douglas C. Schmidt, John Slaby,and Steve Baker, “Domain-Specific Modeling Languages for Configuring and Evaluating Enterprise DRE System Quality of Service, Vanderbilt University Technical Report #ISIS-05-606, August 2005.

TR11 John M. Slaby, Steve Baker, James Hill, Doug Schmidt, “Applying System Execution Modeling Tools to Evaluate Enterprise Distributed Real-time and Embedded System QoS,” Vanderbilt University Technical Report #ISIS-05-604, June 2005.

TR10 Fred Kuhns and Carlos O’Ryan and Douglas C. Schmidt and Jeff Parsons, “The Design and Performance of a Pluggable Protocols Framework for Object Request Broker Middleware,” Washington University Technical Report #WUCS-99-12, St. Louis, MO, Dept. of Computer Science, April 1999.

TR9 Sumedh Mungee, Nagarajan Surendran, and Douglas C. Schmidt, “The Design and Performance of a CORBA Audio/Video Streaming Service,” Washington University Technical Report #WUCS-98-15.

TR8 Lutz Prechelt, Barbara Unger, Douglas C. Schmidt, “Replication of the First Controlled Experiment on the Usefulness of Design Patterns: Detailed Description and Evaluation.” 77 pgs., Washington University Technical Report #wucs-97-34, December 1997.

TR7 Aniruddha Gokhale and Douglas C. Schmidt, “Optimizing the Performance of the CORBA Internet Inter-ORB Protocol Over ATM,” Washington University Technical Report #WUCS-97-10.

TR6 James Hu and Sumedh Mungee and Douglas C. Schmidt, “Principles for Developing and Measuring High-performance Web Servers over ATM,” Washington University Technical Report #WUCS-97-10.

TR5 Chris Cleeland, Douglas C. Schmidt, and Tim H. Harrison, “External Polymorphism – An Object Structural Pattern for Transparently Extending Concrete Data Types,” The 3rd annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 4-6, 1996, Washington University Technical Report #WUCS-97-07.

TR4 Timothy H. Harrison, Douglas C. Schmidt, and Irfan Pyarali, “Asynchronous Completion Token,” The 3rd annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 4-6, 1996, Washington University Technical Report #WUCS-97-07.

TR3 Douglas C. Schmidt and Timothy H. Harrison, “The Double-Checked Locking Pattern,” The 3rd annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 4-6, 1996, Washington University Technical Report #WUCS-97-07.

TR2 Prashant Jain and Douglas C. Schmidt, “The Service Configurator Pattern,” The 3rd annual Pattern Languages of Programming conference in Allerton Park, Illinois, September 4-6, 1996, Washington University Technical Report #WUCS-97-07.

TR1 Douglas C. Schmidt, "Acceptor and Connector: Design Patterns for Initializing Network Services," The EuroPLoP '96 conference in Kloster Irsee, Germany, July 10-14, 1996, Washington University Technical Report #WUCS-97-07.

Presentations

Conference Presentations

1. "Mobile Applications Technology Overview," Digital Technologies in Cancer Research Workshop, Vanderbilt University, Nashville, TN, May 15th 2019.
2. "Website Applications Technology Overview," Digital Technologies in Cancer Research Workshop, Vanderbilt University, Nashville, TN, May 15th 2019.
3. "Producing and Delivering a Coursera MOOC on Pattern-Oriented Software Architecture for Concurrent and Networked Software," WaveFront forum at the SPLASH 2013 conference, Indianapolis, IN, October 29th, 2013.
4. "Addressing the Challenges of Tactical Information Management in Net-Centric Systems with the OMG Data Distribution Service (DDS)," the 16th International ACM Workshop on Parallel and Distributed Real-Time Systems (WPDRTS '08), Miami, Florida, April 14, 2008.
5. "The Design and Performance of Configurable Component Middleware for End-to-End Adaptation of Distributed Real-time Embedded Systems," proceedings of the 10th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC), May 7-9, 2007, Santorini Island, Greece.
6. "A Decision-Theoretic Planner for DRE Systems," 7th OMG Real-time/Embedded CORBA workshop, Arlington, VA, July 10–13, 2006.
7. "Model-driven QoS Provisioning for Real-time CORBA and CCM DRE Systems," 6th OMG Real-time/Embedded CORBA workshop, Arlington, VA, July 11–14, 2005.
8. "Research Advances in Middleware for Distributed Systems: State of the Art," Computer Communications stream of the 17th IFIP World Computer Congress, Montreal, Canada, August 25-30, 2002.
9. "Towards Highly Configurable Real-time Object Request Brokers," the IEEE International Symposium on Object-Oriented Real-time Distributed Computing (ISORC), Washington DC, April 29 – May 1, 2002.
10. "Operating System Performance in Support of Real-time Middleware," Proceedings of the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA, January, 2002.
11. "Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA," Proceedings of the ACM SIGPLAN Workshop on Optimization of Middleware and Distributed Systems (OM 2001), Snowbird, Utah, June 18, 2001.
12. "DOORS: Towards High-performance Fault-Tolerant CORBA," Proceedings of the 2nd International Symposium on Distributed Objects and Applications (DOA '00), OMG, Antwerp, Belgium, September 2000.
13. "The Design and Performance of a CORBA Audio/Video Streaming Service," Proceedings of the 31st Hawaii International Conference on System Systems (HICSS), Hawaii, January, 1999, minitrack on Multimedia DBMS and the WWW, Hawaii, January 1999.
14. "Alleviating Priority Inversion and Non-determinism in Real-time CORBA ORB Core Architectures," Proceedings of the Fourth IEEE Real-Time Technology and Applications Symposium (RTAS), Denver, Colorado, June 3-5, 1998
15. "Optimizing the Performance of the CORBA Internet Inter-ORB Protocol Over ATM," Proceedings of the 31st Hawaii International Conference on System Systems (HICSS), Hawaii, January, 1998. This was selected as the best paper in the Software Technology Track (188 submitted, 77 accepted).
16. "The Double-Checked Locking Pattern," *Proceedings of the 3rd annual Pattern Languages of Programming conference* in Allerton Park, Illinois, September 4-6, 1996.

17. "Acceptor and Connector: Design Patterns for Initializing Network Services," Proceedings of the EuroPLoP '96 conference, Kloster Irsee, Germany, July 10-14, 1996.
18. "Measuring the Performance of Communication Middleware on High-Speed Networks," Proceedings of SIGCOMM '96, ACM, San Francisco, August 28-30th, 1996.
19. "Design and Performance of an Object-Oriented Framework for High-Speed Electronic Medical Imaging," Proceedings of the 2^{nd} Conference on Object-Oriented Technologies and Systems (COOTS), USENIX, Toronto, June 18-22, 1996.
20. "A Family of Design Patterns For Flexibly Configuring Network Services in Distributed Systems," Proceedings of the International Conference on Configurable Distributed Systems, IEEE, Annapolis, Maryland, May 6-8, 1996.
21. "Using Design Patterns to Develop High-Performance Object-Oriented Communication Software Frameworks," Proceedings of the 8^{th} Annual Software Technology Conference, Salt Lake City, Utah, April 21-26, 1996.
22. "An Object-Oriented Framework for High-Performance Electronic Medical Imaging," Proceedings of the *Very High Resolution and Quality Imaging* mini-conference at the Symposium on Electronic Imaging in the International Symposia Photonics West 1996, SPIE, San Jose, California USA, January 27 - February 2, 1996.
23. "Half-Sync/Half-Async: A Pattern for Efficient and Well-structured Concurrent I/O," *Proceedings of the 2^{nd} Pattern Languages of Programs Conference* Monticello, Illinois, September 6-8, 1995.
24. "Acceptor and Connector: Design Patterns for Actively and Passively Initializing Network Services." Workshop on Pattern Languages of Object-Oriented Programs at ECOOP '95, August 7 – 1, 1995, Aarhus, Denmark.
25. "Object-Oriented Components for High-speed Network Programming," *Proceedings of the Conference on Object-Oriented Technologies (COOTS)*, USENIX, June 26-29, 1995 Monterey, California, USA, pp. 21–38.
26. "Experience Using Design Patterns to Evolve Communication Software Across Diverse OS Platforms," *Proceedings of the 9^{th} European Conference on Object-Oriented Programming (ECOOP)*, ACM, Aarhus, Denmark, August, 1995,.
27. "Measuring the Performance of Parallel Message-based Process Architectures," *Proceedings of the INFOCOM Conference on Computer Communications*, IEEE, Boston, MA, April, 1995, pp. 624–633.
28. "High-Performance Event Filtering for Dynamic Multi-point Applications," Proceedings of the 1^{st} Workshop on High Performance Protocol Architectures (HIPPARCH), INRIA, Sophia Antipolis, France, December, 1994, p 1–8.
29. "Flexible Configuration of High-Performance Object-Oriented Distributed Communication Systems," 9^{th} *OOPSLA Conference, invited paper to the Workshop on Flexibility in Systems Software*, ACM, Portland, Oregon, October, 1994, pp. 1–4.
30. "Performance Experiments on Alternative Methods for Structuring Active Objects in High-Performance Parallel Communication Systems," 9^{th} *OOPSLA Conference, poster session*, ACM, Portland, Oregon, October, 1994, pp. 1–12.
31. "Measuring the Impact of Alternative Parallel Process Architectures on Communication Subsystem Performance," *Proceedings of the Proceedings of the 4^{th} International Workshop on Protocols for High-Speed Networks*, IFIP, Vancouver, British Columbia, August, 1994, pp. 103–118.
32. "Reactor: An Object Behavioral Pattern for Concurrent Event Demultiplexing and Dispatching," *Proceedings of the 1^{st} Annual Conference on the Pattern Languages of Programs*, Monticello, Illinois, August, 1994, pp. 1–10.
33. "Experiences with an Object-Oriented Architecture for Developing Dynamically Extensible Network Management Software," *Proceedings of the Globecom Conference*, IEEE, San Francisco, California, November, 1994, pp. 1–7.
34. "Configuring Function-based Communication Protocols for Distributed Applications," *Proceedings of the 8^{th} International Working Conference on Upper Layer Protocols, Architectures, and Applications*, IFIP, Barcelona, Spain, June 1-3, 1994, pp. 361–376.

35. "The ADAPTIVE Service Executive: An Object-Oriented Architecture for Configuring Concurrent Distributed Communication Systems," *Proceedings of the* 8^{th} *International Working Conference on Upper Layer Protocols, Architectures, and Applications*, IFIP, Barcelona, Spain, June 1-3, 1994, pp. 163–178.

36. "ASX: An Object-Oriented Framework for Developing Distributed Applications," *Proceedings of the* 6^{th} *C++ Conference*, USENIX, Cambridge, Massachusetts, April, 1994, pp. 200–220.

37. "The Service Configurator Framework: An Extensible Architecture for Dynamically Configuring Concurrent, Multi-service Network Daemons," *Proceedings of the* 2^{nd} *International Workshop on Configurable Distributed Systems*, IEEE, Pittsburgh, PA, March 21-23, 1994, pp. 190–201.

38. "Tools for Generating Application-Tailored Multimedia Protocols on Heterogeneous Multi-Processor Platforms," *Proceedings of the* 2^{nd} *Workshop on High-Performance Communications Subsystems*, IEEE, Williamsburg, Virginia, September 1-3, 1993, pp. 1–7.

39. "A Framework for Developing and Experimenting with Parallel Process Architectures to Support High-Performance Transport Systems," *Proceedings of the* 2^{nd} *Workshop on High-Performance Communications Subsystems*, IEEE, Williamsburg, Virginia, September 1-3, 1993, pp. 1–8.

40. "ADAPTIVE: a Framework for Experimenting with High-Performance Transport System Process Architectures," *Proceedings of the* 2^{nd} *International Conference on Computer Communications and Networks*, ISCA, San Diego, California, June 28-30, 1993, pp. 1–8.

41. "ADAPTIVE: A Flexible and Adaptive Transport System Architecture to Support Lightweight Protocols for Multimedia Applications on High-Speed Networks," *Proceedings of the* 1^{st} *Symposium on High Performance Distributed Computing*, IEEE, Syracuse, New York, September 9-11, 1992, pp. 174–186.

42. "GPERF: A Perfect Hash Function Generator," *Proceedings of the* 2^{nd} *C++ Conference*, USENIX, San Francisco, California, April 9-11, 1990, pp. 87–102.

Invited Talks

1. "Architecting the Systems of the Future: A Research Agenda," invited keynote talk at the Doctoral Symposium for the 22nd ACM/IFIP International Conference on Middleware, December 6th, 2021.
2. "Cyber- and Physical-Security Risks," Southern Illinois University course on Domestic Terrorism, July 22nd, 2021.
3. "Architecting the Future of Software Engineering," invited keynote talk at the 16th International Conference on Software Technologies, July 8th, 2021.
4. "Challenges of Certifying Adaptive Dynamic Computing Environments," ARLIS Workshop, January 28th, 2021.
5. "Cyber-Security and You: Practicing Safe Surfing on the Internet," the National Active and Retired Federal Employees (NARFE) chapter, Nashville TN, January 13th, 2021.
6. "Challenges of Certifying Adaptive Dynamic Computing Environments," DARPA/SEI Software Engineering Grand Challenges and Future Visions Workshop, December 1st, 2020.
7. "Survelliance Capitalism and You," invited talk at the Southeast Science Boot Camp, Nashville, TN, May 29th, 2019.
8. "Diversify Sponsorship of Your Research: Getting Funding from the Department of Defense," Office of Research Development and Support Workshop, October 22nd, 2018, Nashville, TN.
9. "Survelliance Capitalism and You," invited talk at the Memorizing the Future: Collecting in the 21st Century Conference, Nashville, TN, October 6th, 2018.
10. "Aligning Incentives to Enable More Effective Organic Software Infrastructure for the DoD," DoD Organic Software Infrastructure Workshop, Arlington VA, August 13th, 2018.
11. "The Blockchain: What It is and Why It Matters to Us," Transforming Dermatology in the Digital Era, Memorial Sloan Kettering Cancer Center, October 25, 2018, NY, NY, USA.
12. "Aligning Incentives to Enable Modular Open Software for DoD Combat Systems," Modular Open Systems Summit, May 1st, 2018, Washington DC.

13. "The Blockchain: What It is and Why It Matters to Us," Society of Women Engineers, Vanderbilt University, March 14th, 2018.
14. "The Blockchain: What It is and Why It Matters to Us," Invited keynote at the Workshop on Middleware and Applications for the Internet of Things, (co-located with the 2017 Middleware conference in Las Vegas, USA), December 11th and 12th, 2017.
15. "The Blockchain: What It is and Why It Matters," Vanderbilt University, Nashville, TN, November 28th, 2017.
16. "The Blockchain: What It is and Why It Matters," INTERFACE Nashville conference, Nashville, TN, August 24th, 2017.
17. "Applying Blockchain to Healthcare Systems," panel presentation at the Siemens Blockchain Conference, Nuremburg, Germany, May 10th, 2017.
18. "A Primer on Big Data," Vanderbilt University Board of Trust meeting, April 21st, 2017, Nashville TN.
19. "The Past, Present, and Future of MOOCs and Their Importance for Software Engineering," Distinguished Lecture, University of Illinois Chicago, Chicago, IL, November 18th, 2016.
20. "Agility-at-Scale for Safety- and Mission-Critical Industrial-Scale Systems," INFORMS Annual Conference, Nashville, TN November 13th, 2016.
21. "Product Line Architectures for Open System Architectures," Varian, Winnipeg, Canada, October 7th, 2016.
22. "Agility-at-Scale for Safety- and Mission-Critical Industrial-Scale Systems," Siemens Architecture Workshop, Tarrytown, NY, September 27th, 2016.
23. "Product Line Architectures for Oncology Treatment Services," Varian, Palo Alto, CA, September 16th, 2016.
24. "Innovation and Speed: The Rise of Open Systems," the United States Technology Leadership Council, Reston, VA, August 24th, 2016.
25. "Elastic Software Infrastructure to Support the Industrial Internet," the Siemens CPS Workshop, Munich, Germany, August 1st, 2016.
26. "Challenges of Certifying Adaptive Dynamic Computing Environments," Workshop on Safety And Control for AI, Sponsored by the White House Office of Science and Technology Policy and Carnegie Mellon University, Pittsburgh, PA, June 28th, 2016.
27. "Keeping an Unfair Advantage in a Globalized and Commoditized World," Raytheon Symposium, Tucson, AZ, May 5th, 2016.
28. "Towards Technical Reference Frameworks to Support Open System Architecture Initiatives," Office of the Secretary of Defense (OSD) System of Systems Engineering Collaborators Information Exchange, December 15th 2015.
29. "Enterprise System of Systems Engineering (SoSE) Integration and Innovation," presentation at the US Marine Corp Business Management Association meeting, Quantico, VA, December 10th, 2015.
30. "An Architecture Grand Challenge: DOD's push for Open Systems Architecture," panel presentation at the Software Solutions Conference, Crystal City, VA, November 17th, 2015.
31. "Elastic Software Infrastructure to Support the Industrial Internet," the Siemens CPS Workshop, Munich, Germany, September 29th, 2015.
32. "An Overview of Mobile and mHealth Activities at ISIS and Vandy EECS," Patient Engagement Emerging Technologies, Vanderbilt University, Nashville, TN, August 10, 2015.
33. "Mobile Cloud Computing with Android," Google I/O, Mercury Intermedia Systems, Nashville, TN, May 28th, 2015.
34. "An Architecture Grand Challenge: DOD's push for Open Systems Architecture," panel presentation at the SATURN 2015 Conference, Baltimore, MD, April 27th, 2015.
35. "'Elastic Software Infrastructure to Support Computing Clouds for Cyber-Physical Systems," Distinguish Lecture, Texas A&M, April 27th, 2015.

36. “Elastic Software Infrastructure to Support Computing Clouds for Cyber-Physical Systems”, Boeing Distinguished Researcher And Scholar Seminar (B-DRASS) series, March 20th, Huntingon Beach, CA.
37. “Elastic Software Infrastructure to Support Computing Clouds for Cyber-Physical Systems,” Distinguished Lecture, University of California, Irvine, February 27th, 2015.
38. “Elastic Software Infrastructure to Support Computing Clouds for Cyber-Physical Systems,” Varian, Palo Alto, CA, January 15th, 2015.
39. “Keeping an Unfair Advantage in a Globalized and Commoditized World,” Open Architecture Summit, Washington DC, November 4th, 2014.
40. “Proposal for a Professional Masters degree in Computer Science,” invited talk at Vanderbilt University School of Engineering’s Board of Visitor’s meeting, October 10th, 2014.
41. “The Past, Present, and Future of Open System Architecture Initiatives,” invited keynote at the Future Airborne Capabilities Environment meeting, Nashville, TN, September 24th.
42. “Future Proofing Research and Development Investments in a Globalized, Commoditized World,” Boeing Technical Excellence Conference, May 20th, 2014, St. Louis, MO.
43. “Elastic Software Infrastructure to Support the Computing Clouds for Cyber-Physical Systems (CC4CPS),” Securboration Conference, November 12th, 2013, Melbourne, Florida.
44. “The Importance of Automated Testing in Open Systems Architecture Initiatives,” Open Architecture Summit, November 12th, 2013, Washington DC.
45. Conference on Cloud and Mobile Computing, Siemens Corporate Research, Princeton, NJ, November 5th, 2013.
46. “Overview of the Technology Entrepreneurship Task Force,” Innovation, Imagination, and Introductions: A Conversation with Entrepreneurs, Vanderbilt University, October 24th, 2013.
47. “Producing and Delivering a Coursera MOOC on Pattern-Oriented Software Architecture for Concurrent and Networked Software,” Vanderbilt University’s Faculty Senate committee on Strategic Planning and Academic Freedom, October 23rd, 2013.
48. “Elastic Software Infrastructure to Support the Industrial Internet,” RTI Webinar series, October 23rd, 2013.
49. “The Importance of Applying Agility to DoD Software Initiatives,” IEEE Computer Society Lockheed Martin webinar series, October 10th, 2013.
50. “Technology Entrepreneurship Task force: Charter and Progress Update,” Vanderbilt University School of Engineering Board of Visitors meeting, October 4th, 2013.
51. “Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation USAF Combat Systems,” AFOSR DDDAS PI meeting, Arlington, VA, October 1st, 2013.
52. “Producing and Delivering a Coursera MOOC on Pattern-Oriented Software Architecture for Concurrent and Networked Software,” WithIT seminar, Vanderbilt University, September 12th, 2013.
53. “Applying Agility to the US Department of Defense Common Operating Platform Environment Initiatives,” Interoperable Open Architecture conference, Washington DC, September 11th, 2013.
54. “Software Infrastructure Support of Computing Clouds for Cyber-Physical Systems,” invited talk at Real-Time Innovations, July 31st, 2013, Sunnyvale, California.
55. “Introduction to the Institute for Software Integrated Systems,” Nashville Entrepreneur Center, July 15th, 2013.
56. ”Surviving the Coursera Digital Learning Experience,” Coursera-in-TN Conference, Vanderbilt University, Nashville, TN, June 24th, 2013.
57. “Quo Vadis ISORC?,” Panel presentation at ISORC 2013 Conference, June 19th, 2013, Paderborn, Germany.
58. “Software Infrastructure Support of Computing Clouds for Cyber-Physical Systems,” invited keynote for ISORC 2013 Conference, June 19th, 2013, Paderborn, Germany.
59. “Towards Programming Models and Paradigms for Computing Clouds that Support Cyber-Physical Systems,” NSF Workshop on Computing Clouds for Cyber-Physical Systems, March 15th, 2013, Ballston, VA.

60. “Built to Last: Planning Your Career as an Engineer,” STEM contest on Securing Cyber Space, Brentwood High School, March 9th, 2013, Nashville, TN.
61. “Experience with Digital Learning and MOOCs at Vanderbilt,” Nashville, TN, Feb 22nd, 2013.
62. “Software Design: Is It Really Better to Look Good Than to Feel Good?,” World IA Day, Nashville, TN, Feb 9th, 2013.
63. “Pattern-Oriented Software Architectures: Patterns and Frameworks for Concurrent and Networked Software,” PhreakNIC 2012, Murfreesboro, TN, November 9th, 2012.
64. “Applying Agility to the US Department of Defence Common Operating Platform Environment Initiatives,” Interoperable Open Architecture 2012, 29 - 31 October, 2012, London, UK.
65. “Open System Architectures: Challenges and Success Drivers,” OA Summit conference, Washington, DC, October 18th, 2012.
66. “Dependable Computing Clouds for Cyber-Physical Systems,” Dependability Issues in Cloud Computing Workshop, October 11th, 2012, Irvine, CA.
67. “Computing Clouds for Cyber-Physical Systems,” Reliable Cloud Infrastructure for CPS Applications Workshop, October 8th, 2012, Irvine, CA.
68. “Common Operating Platform Environments: Challenges and Success Drivers,” Navy Open Systems Architecture workshop, Ballston, VA, September 27th, 2012.
69. ”Meeting the Challenges of Enterprise Distributed Real-time and Embedded Systems,” talk for Honeywell Aerospace, September 21, 2012.
70. “Architecture-Led Iterative and Incremental Development for Common Operating Platform Environments,” NITRD Software Design and Productivity meeting, National Coordination Office, Ballston, VA, July 13th, 2012.
71. ”Cyber-physical multi-core Optimization for Resource and cachE effectS,” Software-Intensive Systems Producibility workshop, Arlington VA, June 5th, 2012.
72. ”Applying Agility to DoD Common Operating Platform Environment Initiatives”, SEI Agile Research Forum, May 22nd, 2012.
73. “Meeting the Challenges of Enterprise Distributed Real-time and Embedded Systems,” keynote talk at the SATURN Conference 2012 May 7-11, 2012, St. Petersburg, FL.
74. “Reflections on 20 Years of Architecture for Distributed Real-time and Embedded Systems,” SATURN Conference 2012 May 7-11, 2012, St. Petersburg, FL.
75. “US Naval Open Systems Architecture Strategy,” SATURN Conference 2012 May 7-11, 2012, St. Petersburg, FL.
76. ”Towards Open Systems Architectures for Distributed Real-time and Embedded Systems,” The Center for Embedded Systems for Critical Applications, Annual Workshop, Virginia Tech, Blackburg, VA April 21st, 2012.
77. “Overview of the SEI Strategic Research Plan,” ASD(R&E) Annual Program Review, December 7th, 2011, Pittsburgh, PA.
78. “Overview of the SEI Strategic Research Plan,” Acquisition Support Program meeting, November 16th, 2011, Pittsburgh, PA.
79. “Conducting Leading-Edge Software R&D in a Globalized, Commoditized World,” NITRD Software Design and Productivity meeting, National Coordination Office, Ballston, VA, November 3rd, 2011.
80. “A Technical Assessment of Open Architecture Systems for Military Use,” Interoperable Open Architecture, 26th-28th October 2011, London, UK.
81. “Conducting Leading-Edge Software R&D in a Globalized, Commoditized World,” Technovation 2011, Carnegie Mellon University, September 29th, 2011.
82. “CTO Report,” SEI Board of Visitors Meeting, Arlington, VA, September 27th, 2011.
83. “Overview of the SEI Strategic Research Plan,” Joint Advisory Committee Meeting, Arlington, VA, September 26th, 2011.

84. "Successful Development Efforts: Standards, People, & Culture: The Enterprise Perspective," Software Assurance (SwA) Forum, September 16th, 2011, Arlington, VA.
85. "Ultra-Large-Scale (ULS) Cyberphysical Systems and Their Impact on Technology and Society," University of Salzburg, June 30th, 2011, Salzburg, Austria.
86. "Ultra-Large-Scale (ULS) Cyberphysical Systems and Their Impact on Technology and Society," ARTEMIS conference, June 29th, 2011, Linz, Austria.
87. "Ultra-Large-Scale Systems and Their Impact on the DoD," Systems and Software Technology Conference Committee, keynote presentation at the 23rd Systems and Software Technology Conference, May 16-19, 2011, Salt Lake City, Utah.
88. "Ultra-Large Scale Systems and their Impact on Technology and Society," keynote presentation at the International Symposium on Object-Oriented Real-time Distributed Computing¡/A¿ (ISORC), Newport Beach, CA, March 29th, 2011.
89. "Software-reliant Systems Research at the Software Engineering Institute," Raytheon, Sudbury, MA, March 10, 2011.
90. "Review of COE Practices," US Army Senior Leadership Education Program, Pittsburgh, PA, January 20th, 2011.
91. "Software Producibility for Defense," US Army Senior Leadership Education Program, Pittsburgh, PA, January 18th, 2011.
92. "SEI Research: The Shape of Things to Come," ASP Meeting, Software Engineering Institute, Pittsburgh, PA, December 9th, 2010.
93. "R&D at ASP," ASP Air Force Training Day, Software Engineering Institute, Pittsburgh, PA, December 9th, 2010.
94. "Software-reliant Systems Research at the Software Engineering Institute," Siements Corporate Research, Princeton, NJ, November 22nd, 2010.
95. "Taming the Complexity of Software-Reliant Systems," Software Engineering Process Group conference, Colombia, South America, November 11th, 2010.
96. "SEI Technical Presentations," Joint Advisory Committee Meeting, Arlington, VA, October 26th, 2010.
97. "SEI Research: The Shape of Things to Come," ASP Meeting, Software Engineering Institute, Pittsburgh, PA, October 20th, 2010.
98. "SEI Research: The Shape of Things to Come," SEPM Meeting, Software Engineering Institute, Pittsburgh, PA, October 19th, 2010.
99. "Strategic Directions for Research at the SEI," RTSS Offsite Meeting, Pittsburgh, PA, October 12th, 2010.
100. "The World is Flat and What You Can Do About It," Family Weekend, October 9th, 2010, Vanderbilt University.
101. "SEI Research: The Shape of Things to Come," SEI Board of Visitor's Meeting, Arlington, VA, September 28th, 2010.
102. "SEI Research: The Shape of Things to Come," PD&T Meeting, Software Engineering Institute, Pittsburgh, PA, September 20th, 2010.
103. "Introduction and Initial Thoughts," RTSS Meeting, Software Engineering Institute, Pittsburgh, PA, August 19th, 2010.
104. "The Impact of Ultra-Large-Scale Systems on DoD Operations," Congressional R&D Caucus, Rayburn Building, Washington DC, January 19th, 2010.
105. "The World is Flat and What You Can Do About It," Explorers meeting, January 12th, 2010, Vanderbilt University.
106. "Expectations for University - Industry Collaborative Research in CPS," Computing Community Consortium Workshop on New Forms of Industry-Academy Partnerships in CPS Research, George Mason University, May 19th, 2009.
107. "How Good is Your SOA?", Panel presentation at the AFRL QED PI meeting, April 28th, 2009, Washington DC.

108. "The World is Flat and What You Can Do About It," ES 140, Computer Science module, October 31st, 2008, Vanderbilt University.
109. "Meeting the Challenges of Ultra-Large-Scale Distributed Real-time and Embedded Systems with QoS-enabled Middleware and Model-Driven Engineering," Panel on Growing and Sustaining Ultra Large Scale (ULS) Systems, OOPSLA 2008, Nashville TN, October 21-23 2008.
110. "The World is Flat and What You Can Do About It," Family Weekend Faculty Lecture, Vanderbilt University, October 3rd, 2008.
111. "The World is Flat and What You Can Do About It," Senior Design Seminar, Vanderbilt University, September 17th, 2008.
112. "The World is Flat and What You Can Do About It," CS WithIT Seminar, Vanderbilt University, September 11th, 2008.
113. "The Managed Motorway: Real-time Vehicle Scheduling - A Research Agenda," Qualcomm, July 28th, 2008, San Diego, CA.
114. "Meeting the Challenges of Mission-Critical Distributed Event-Based Systems with QoS-enabled Middleware and Model-Driven Engineering," 2nd International Conference on Distributed Event-Based Systems (DEBS), Rome Italy, July 2-4, 2008.
115. "Meeting the Challenges of Distributed Real-time and Embedded Systems with QoS-enabled Middleware and Model-Driven Engineering," SPAWAR, April 29th, 2008.
116. "Meeting the Challenges of Distributed Real-time and Embedded Systems with QoS-enabled Middleware and Model-Driven Engineering," Northrop Grumman, Boulder Colorado, April 25th, 2008.
117. "Experimentation Environment for QED," AFRL Information Management PI Meeting, April 16 2008, Georgetown, Washington, DC.
118. "Adaptive System Infrastructure for Ultra-Large-Scale Systems," SMART Conference, Carnegie Mellon University, March 6th, 2008.
119. "Experimentation Environment for QED", Air Force Research Lab, Rome, NY, March 4th, 2008.
120. "Ultra-Large-Scale (ULS) Systems and their Impact on Technology and Society," Clemson University, January 31st, 2008.
121. "Meeting the Challenges of Ultra-Large-Scale Distributed Real-time and Embedded Systems with QoS-enabled Middleware and Model-Driven Engineering, invited keynote talk at Middleware 2007, Irvine, CA, November 29th, 2007.
122. "The World is Flat and What You Can Do About It," Senior Design Seminar, Vanderbilt University, November 14th, 2007.
123. "Technology Candidates for QED," AFRL retreat, Minnowbrook, NY, October 23, 2007.
124. "Overview of ISIS and Proposed IU/CRC R&D Projects," Crystal City, VA, October 19th, 2007.
125. The Future of CORBA for Distributed Real-time and Embedded Systems, International Conference on Accelerator and Large Experimental Physics Control Systems, October 17, 2007, Knoxville, TN.
126. "AF-TRUST: Project Overview," Air Force Scientific Advisory Board review, Rome, NY, October 15th, 2007.
127. "Meeting the Challenges of Distributed Real-time and Embedded Systems with Product-Line Architectures," August 1st, 2007, Trinity College, Dublin, Ireland.
128. "Model Driven Engineering of Product-Line Architectures for Distributed Real-time and Embedded Systems," July 5th, 2007, University of Limerick, Ireland.
129. "Meeting the Challenges of Mission-Critical Systems with Middleware and Model Driven Engineering", OMG Technical Meeting, June 27, 2007, Brussels, Belgium.
130. Meeting the Challenges of Ultra-Large-Scale Distributed Real-time and Embedded Systems with Model-Driven Engineering, June 19, 2007, Trinity College, Dublin.
131. Strategic Technology Positioning, PrismTechnologies "Middleware Fest", June 14, 2007, Newcastle, UK.
132. "Hurdles for Wireless Communication Systems R&D and Some Ways to Overcome Them," OSD Workshop on Wireless Communication Systems, Rosslyn, VA, May 22nd, 2007.

133. “The World is Flat from a Computer Scientists Point of View,” Vanderbilt University Commencement talk, May 10th, 2007.
134. Meeting the Challenges of Ultra-Large-Scale Distributed Real-time and Embedded Systems, invited keynote at the the 10th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing, May 7-9, 2007, Santorini Island, Greece.
135. “Enhanced QoS for the GIG,” AFRL JBI PI meeting, Georgetown, DC, April 24, 2007.
136. “Meeting the Challenges of Ultra-Large-Scale Distributed Real-time and Embedded Systems,” Invited keynote at the 15th International Workshop on Parallel and Distributed Real-Time Systems (WDPRTS), March 26-27, 2007, Long Beach, California.
137. “The CORBA C++ Mapping: Beyond Repair?,” OMG Meeting, San Diego, CA, March 27th, 2007.
138. “Meeting the Challenges of Ultra-Large-Scale Systems via Model-Driven Engineering,” Distinguished Lecturer Series, Florida International University, Miami, Florida, Feb 2, 2007.
139. Model Driven Engineering and QoS-enabled Component Middleware for DRE Systems, Invited talk at the European Space Agency Operations Center, Darmstadt, Germany, Wednesday January 24, 2007.
140. ”Software Wind Tunnel (SWiT) Concept of Operations and System Architecture”, AFRL Software and Systems Test Track workshop, Arlington, VA, January 19, 2007.
141. “Latest Breakthroughs in SDR Software Development Using Model Driven Technologies,” Rockwell Collins, Cedar Rapids, IA, December 14th, 2006.
142. “Educating the DoD Workforce in a Flat World,” 2006 Raytheon Integrated Defense Systems’ SW Engr. Directorate Off-Site Meeting, New Castle, New Hampshire, December 7, 2006.
143. “The Ultra Challenge: Software Systems Beyond Big,” panelist at OOPSLA 2006, October, 2006, Portland, OR.
144. “Software Wind Tunnel (SWiT) Architecture,” AFRL Software and Systems Test Track Workshop, Cherry Hill, NJ, October 2nd, 2006.
145. “The World is Flat and What You Can Do About it,” Vanderbilt University, September 12th, 2006.
146. “The World is Flat and What You Can Do About it,” Vanderbilt University, September 8th, 2006.
147. “Meeting the Challenges of Ultra-Large-Scale Systems via Model-Driven Engineering,” Network-Centric Operations Industry Consortium, Reston, VA, August 2nd 2006.
148. Model Driven Architecture Roundtable, invited panelist at the Software Engineering Institute, Pittsburgh, PA, June 1st, 2006.
149. “Enhanced QoS for the GIG,” AFRL JBI PI meeting, Tysons Corner, VA, April 11, 2006.
150. “Model Driven Engineering for Distributed Real-time and Embedded Systems,” Distinguished Lecturer Series talk at Colorado State University, Ft. Collins, CO, April 10, 2006.
151. ”Win-Win Partnership of Academia and Industry: Why Should We Care? Where Is Our Common Future?” invited panelist at the 12th IEEE Real-Time and Embedded Technology and Applications Symposium April 6, 2006, San Jose, California.
152. “Meeting the Challenges of Ultra-Large-Scale Real-time Systems,” invited keynote at the IEEE Real-Time and Embedded Technology and Applications Symposium April 5, 2006, San Jose, California.
153. “Model-driven Development for Distributed Real-time and Embedded Systems,” ACM Meeting at Middle Tennessee State University, March 7th, 2006.
154. “Real-time, Scalable, and Secure Information Management for the GIG,” Scientific Advisory Board Meeting, Rome, NY, November 16th, 2005.
155. “Real-time, Scalable, and Secure Information Management for the GIG,” Airforce Research Lab, Rome, NY, November 3rd, 2005.
156. “Model-driven Development for Distributed Real-time and Embedded Systems,” Distinguished Speaker Talk at BBN Technologies, Cambridge, MA, October 27, 2005.

157. "Challenges and Research Areas for QoS-enabled Information Management in Tactical Systems of Systems," AFRL Minnowbrook Workshop, Adirondack Mountains, NY, October 21st, 2005.
158. "Model-driven Development for Distributed Real-time and Embedded Systems," Invited keynote at MODELS 2005, ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems, Half Moon Resort, Montego Bay, Jamaica, October 5-7, 2005.
159. "The World is Flat and What You Can Do About it," CS WithIT Seminar, Vanderbilt University, September 22, 2005.
160. "Why Software Reuse has Failed and How to Make it Work for You," Motorola 2005, Symposium on Software, Systems, and Simulation, Schaumburg, IL, September 16th, 2005.
161. "Pattern-Oriented Software Architecture," 12th Pattern Language of Programming Conference, Allerton Park, Illinois, September 7-10, 2005.
162. "Model-Driven Development of Distributed Real-time and Embedded Systems," 12th Pattern Language of Programming Conference, Allerton Park, Illinois, September 7-10, 2005.
163. "Model-driven Development for Distributed Real-time and Embedded Systems," Siemens Corporate Research, Princeton, NJ, August 26th.
164. "Model-driven QoS Provisioning for Real-time CORBA and CCM DRE Systems," 6th OMG Real-time/Embedded CORBA workshop, Washington DC, July 11–14, 2005.
165. "A Proposed R&D Agenda for the Software Technology Laboratory," Lockheed Martin Advanced Technology Lab, Cherry Hill, NJ, June 28th, 2005.
166. "Model-Driven Development of Product-Line Architectures for DRE Systems," 11th Siemens Software Architecture Improvement Group (SAIG), Buffalo Grove, IL June 22, 2005.
167. "Business Drives for Platforms," panel at the 11th Siemens Software Architecture Improvement Group (SAIG), Buffalo Grove, IL June 22, 2005.
168. "Model Driven Development for Distributed Real-time and Embedded Systems," Lockheed Martin Advanced Technology Lab, Cherry Hill, NJ, June 15th, 2005.
169. "Approaches for Supporting Real-time QoS in JBI," JBI PI Meeting, Washington DC, May 24th, 2005.
170. "Overcoming Hurdles of Software Producibility," OSD, Software Producibility Workshop, Arlington, VA, May 18, 2005.
171. "Overview of Multi-Level Resource Management in ARMS," Fermilab, Chicago, IL, April 12th, 2005.
172. "Model Driven Middleware for Distributed Real-time and Embedded Systems," University of Southern Alabama, April 8, 2005.
173. "Model-Driven Development of Distributed Real-time and Embedded Systems," UAV Battlelab, Indian Springs, NV, February 10th, 2005.
174. "The Future of Software and Systems Engineering," IEEE Meeting, Vanderbilt University, February 8th, 2005.
175. Model Driven Development of Distributed Real-time and Embedded Systems, panel at the OOP conference, Munich, Germany, January 27, 2005.
176. "Product-line Architecture Technologies for Distributed Real-time and Embedded Systems, Lockheed Martin, Moorestown, NJ, November 11, 2004.
177. "Model Driven Development of Distributed Real-time and Embedded Systems," invited panelist in the "Generative Programming: Past, Present, and Future," at the 3rd ACM International Conference on Generative Programming and Component Engineering, Vancouver, CA, October 24th 2004.
178. "Developing Combat Systems with Component Middleware and Models," Lockheed Martin, Moorestown, NJ, October 22, 2004.
179. "Model Driven Development of Distributed Real-time and Embedded Systems," Lockheed Martin Advanced Technology Lab, Cherry Hill, NJ, October 21, 2004.
180. "Model Driven Development of Distributed Real-time and Embedded Systems," Lockheed Martin Missile and Fire Control, Dallas, TX, October 13, 2004.

181. "Design of ARMS MLRM Components: CCM Based Design for Dynamic Resource Management," DARPA ARMS Technical Interchange Meeting, Plymouth, RI, October 7, 2004.
182. "Model Driven Middleware for Component-based Distributed Systems," keynote for the The 8th International IEEE Enterprise Distributed Object Computing Conference, Monterey, California, September 22, 2004.
183. "Systems Science Challenge Area," TRUST NSF Science and Technology Review, UC Berkeley, September 12, 2004.
184. "Model Driven Development for Distributed Real-time and Embedded Systems," Lockheed Martin, Eagan, MN, August 31st, 2004.
185. "Model Driven Computing for Distributed Real-time and Embedded Systems," Telcordia, Piscataway, NJ, August 10th, 2004.
186. "Model Driven Computing for Distributed Real-time and Embedded Systems," Raytheon, Portsmouth, RI, August 9th, 2004.
187. "Distributed Object Computing with CORBA," Raytheon, Portsmouth, RI, August 9th, 2004.
188. "Model Driven Development of Distributed Real-time and Embedded Systems," Raytheon, Ft. Wayne, IN, July 27th, 2004.
189. "Model Driven Middleware for Distributed Real-time and Embedded Systems," panelist at the 5th OMG Real-time and Embedded Middleware Workshop, Reston, VA 2004.
190. "The Role of Open Standards,Open-Source Development, and Different Development Models and Processes on Industrializing Software," ARO Workshop on Software Reliability for FCS, Vanderbilt University, Nashville, Tennessee, May 18-19, 2004.
191. "Model Driven Middleware for Distributed Real-time and Embedded Systems," Keynote talk for the SIGS Software Engineering Today conference in Zurich, Switzerland, May 4-5, 2004.
192. "Model-Driven Development of Distributed Real-time and Embedded Systems," 10th Siemens Software Architecture Improvement Group (SAIG), Vienna, Austria, April 20-24, 2004.
193. "Adaptive and Reflective Middleware for Distributed, Real-time, and Embedded Systems," Purdue University, West Lafayette, Indiana, April 6, 2004.
194. "Model Driven Middleware for Distributed Real-time and Embedded Systems," *Technologies That Will Change the World* session at the Southeastern Software Engineering Conference, Huntsville, Alabama, March 30th, 2004.
195. "Advances in COTS Middleware for Distributed Real-time and Embedded Systems," Keynote for the International Conference on COTS-Based Software Systems (ICCBSS) 2004 in Redondo Beach, February 2-4, 2004.
196. Composable Middleware Components for High Confidence Network Embedded Systems, University of California, Berkeley, December 4th, 2003.
197. "Model Driven Middleware," TechConnect 2003, St. Louis, MO, October 1st, 2003.
198. "Advances in Model Driven Middleware for Distributed Real-time and Embedded Systems," the Model Integrated Computing PSIG meeting at the OMG Technical Meeting, September 10, 2003, Boston, MA.
199. Invited panelist for the "Research on DRE Systems" panel at the OMG Real-time Middleware Workshop, July 16, 2003, Arlington, VA.
200. "Advances in Model Driven Middleware for Distributed Real-time and Embedded Systems," the OMG Real-time Middleware Workshop, July 15, 2003, Arlington, VA.
201. Organizer and presenter for a panel on "Advances in Large-scale Distributed Real-time and Embedded Systems" at the 9th IEEE Real-time/Embedded Technology and Applications Symposium (RTAS), May 27-30, 2003, Washington, DC.
202. "Managing Project Risk for Combat Systems," The Southeastern Software Engineering Conference, Huntsville, Alabama, April 1st, 2003.
203. "Distributed Real-time and Embedded Systems at DARPA," OMG Workshop on Super Distributed Objects, Washington DC, Monday, November 18, 2002.

204. “Adaptive and Reflective Middleware for Distributed Real-time Systems,” Workshop on High Performance, Fault Adaptive, Large Scale Real-time Systems, Vanderbilt University, November 14, 2002.
205. Invited panelist on ”Objects and Real-time Systems” OOPSLA ’02, Seattle, WA, November 8, 2002.
206. “An Overview of ACE+TAO,” Boeing, Seattle, November 8th, 2002.
207. “Pattern-Oriented Software Architecture,” Amazon, Seattle, WA, November 6th, 2002.
208. “Using Real-time CORBA Effectively: Patterns and Principles,” CORBA Controls Workshop, Grenoble, France, October, 9th, 2002.
209. “Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems,” EMSOFT 2002: Second Workshop on Embedded Software, Grenoble, France, October, 7–9th, 2002.
210. “Designing the Future of Embedded Systems at DARPA IXO,” Keynote talk at the 6th Annual Workshop on High-Performance Embedded Computing (HPEC), September 25, Boston, MA.
211. “Open Distributed Computing Platforms,” NSF/OSTP Workshop on Information Technology Research for Critical Infrastructure Protection, Lansdowne, VA, September 20th, 2002.
212. “Real-time Object-Oriented Middleware,” Distributed Common Ground/Surface System Technical Review Group meeting, Mclean VA, September 19th, 2002.
213. “Research Advances in Middleware for Distributed, Real-time, and Embedded Systems,” Computer Communications stream of the 17th IFIP World Computer Congress, Montreal, Canada, August 25-30, 2002.
214. “DARPA Thrusts in Embedded Computing,” Mercury Computer Systems, Tyngsboro, MA, July 25th, 2002.
215. “Adaptive and Reflective Middleware for Distributed, Real-time, and Embedded Combat Systems,” Boeing Space and Missile Systems, Anaheim, CA, July 9, 2002.
216. “Annual Report on Software Design and Productivity Coordinating Group,” Interagency Working Group, ITR&D Spring Planning Meeting, NSF, Ballston, VA, May 10, 2002.
217. “Real-time CORBA Standardization: Past, Present, and Future,” panelist in the ”Standards Movements in Object-oriented Real-time Computing” panel at the ISORC 2002 Conference, Washington, DC, April 30, 2002.
218. “Towards Adaptive and Reflective Middleware for Distributed Real-time Embedded Systems,” Moderator of the *Distributed, Real-time, and Embedded Middleware for Network-Centric Combat Systems* panel at the Software Technology Conference (STC) in Salt Lake City, Utah, April 29, 2002.
219. “Applying Architectural Patterns to Address Key Challenges of Distributed Software,” Siemens Architecture Interworking Group, Chicago, IL, April 24, 2002.
220. “Towards Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems,” Space and Missile Defense Command, Huntsville, AL, April 22, 2002.
221. “How to Maintain Superiority in the Face of the Commoditization of IT,” tutorial at the UCI CEO Roundtable, Maui, Hawaii, April 12, 2002.
222. “Transformation or Transmogrification? Surviving the Commoditization of IT,” panelist at the UCI CEO Roundtable, Maui, Hawaii, April 11, 2002.
223. “Patterns and Principles of Mission-critical Middleware,” Henry Samueli School of Engineering Research Review, University of California, Irvine, March 14th, 2002.
224. “DARPA: an Agency Overview,” CRA Academic Careers Workshop, Arlington, Virginia, February 10 - 12, 2002.
225. “Towards Adaptive and Reflective Middleware for Distributed, Real-time, and Embedded Systems,” Electrical Engineering and Computer Science Department, Vanderbilt University, January 28th, 2002.
226. “Protecting Critical Cyber Infrastructure from Asymmetric Threats,” panelist at the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA, January 10, 2002.

227. “The Researcher’s Dilemma: When Technology Success Causes Great Communities to Fail (at Mission-oriented R&D Agencies),” Software Design and Productivity Coordinating Group Workshop on New Visions for Software Design and Productivity: Research and Applications, Nashville, TN, December 13-14, 2001.
228. “Towards Adaptive and Reflective Middleware for Mission-Critical Systems,” Computer Science Department, College of William and Mary, September 7th, 2001.
229. “Adaptive and Reflective Middleware Systems,” Lockheed Martin, Moorestown, NJ, August 21st, 2001.
230. “Adaptive and Reflective Middleware Systems,” United Technology Research Center, Hartford, Connecticut, June 28th, 2001.
231. “Adaptive and Reflective Middleware Systems,” Raytheon Annual Processing Systems Technology Network (PSTN) Symposium, Lexington, MA, June 20th, 2001.
232. Invited presenter for the Vendors’ Panel at the OMG 2nd Workshop on Real-time and Embedded Distributed Object Computing, June 4-7, 2001.
233. ”Towards Pattern Languages and QoS-enabled Middleware for Distributed Real-time and Embedded Systems,” DARPA ITO workshop on Embedded Software, Lake Tahoe, NV, October 8-10, 2001.
234. ”TAO, CORBA, and the HLA/RTI”, Keynote talk at the Fifth IEEE International Workshop on Distributed Simulation and Real Time Applications Cincinnati, Ohio, USA August 13-15, 2001.
235. “Patterns and Principles of Middleware for Distributed Real-time and Embedded Systems,” Raytheon, Sudbury, March 29th, 2001.
236. “Adaptive and Reflective Middleware Systems,” Distinguished Lecture at Florida Atlantic University, Boca Raton, FL, March 1st, 2001.
237. “Adaptive and Reflective Middleware for Mission-Critical Distributed and Embedded Systems,” University of Alabama, Birmingham, AL, January 31st, 2001.
238. “Adaptive and Reflective Middleware for Mission-Critical Distributed and Embedded Systems,” Telcordia, Morristown, NJ, November 20th, 2000.
239. “Adaptive and Reflective Middleware for Mission-Critical Distributed and Embedded Systems,” George Mason University, Fairfax, VA, November 20th, 2000.
240. “Adaptive and Reflective Middleware for Mission-Critical Distributed and Embedded Systems,” Lucent CORBA Forum, Naperville, IL, November 17th, 2000.
241. “Putting an ORB on a Diet,” Session on *Performance and QoS of Embedded CORBA ORBs* at the OMG’s Workshop on Embedded Object-Based Systems, January 17-19, 2001.
242. “Adaptive and Reflective Middleware Systems,” Panelist in a session on “Highly Distributed Systems,” at the IEEE Symposium on Applications and the Internet, San Diego, CA, January 10, 2001.
243. “Adaptive and Reflective Middleware Systems,” Panelist at the NSF Networking PI meeting, Irvine California, November 1st, 2000.
244. “Surviving the Tornado: The Best Kept Secrets of R&D Success in the Internet Age,” Keck Observatory, Hawaii, October 9th, 2000.
245. “Adaptive and Reflective Middleware Systems,” BBN Technologies, Boston, MA, September 27th, 2000.
246. “Distributed Application Integration: Myth or Reality?” Keynote talk at 2nd International Symposium on Distributed Objects and Applications (DOA ’00), OMG, Antwerp, Belgium, September 21st, 2000.
247. “Surviving the Tornado: The Best Kept Secrets of R&D Success in the Internet Age,” Keynote talk at 2nd International Symposium on Distributed Objects and Applications (DOA ’00), OMG, Antwerp, Belgium, September 21st, 2000.
248. “High Confidence Adaptive and Reflective Middleware: Fact or Fiction?” Keynote talk for the IFIP Fourth International Conference on Formal Methods for Open Object-Based Distributed Systems, (FMOODS 2000), Stanford University, Stanford, CA, September 7th, 2000.

249. “Adaptive and Reflective Middleware Systems,” Lockheed Martin, Ft. Worth, TX, September 6th, 2000.
250. Pattern-oriented Software Architecture: Concurrent and Networked Objects, Raytheon, San Diego, August 25, 2000.
251. “Adaptive and Reflective Middleware Systems,” Rockwell/Collins, Cedar Rapids, Iowa, August 22, 2000.
252. “Adaptive and Reflective Middleware Systems,” Lockheed Martin, Eagan, MN, August 21, 2000.
253. “Adaptive and Reflective Middleware Systems,” Honeywell Technology Center, Minneapolis, MN, August 18, 2000.
254. “Adaptive and Reflective Middleware Systems,” Raytheon, Falls Church, VA, July 12, 2000.
255. “Applying Patterns to Develop High-performance and Real-time Object Request Brokers,” Lockheed Martin, Eagan, Minnesota, May 19, 2000.
256. “Patterns and Principles of Real-time Object Request Brokers,” Cisco, San Jose, April 12, 2000.
257. “Patterns and Principles of Real-time Object Request Brokers,” BellSouth, Atlanta, Georgia, March 3, 2000.
258. “Patterns and Principles of Real-time Object Request Brokers,” Distinguished Lecturer Series, Michigan State University, East Lansing, Michigan, October 21, 1999.
259. “Towards Minimum ORBs for Wireless Devices and Networks,” OPENSIG ’99 Workshop, Carnegie Mellon University, Pittsburgh, October, 14-15, 1999.
260. “Applying CORBA Fault Tolerant Mechanisms to Network Management,” Lucent CORBA Forum, Naperville, IL, September 28th, 1999.
261. “CORBA for Real-time and Embedded Telecom Systems,” Lucent CORBA Forum, Naperville, IL, September 28th, 1999.
262. “Patterns and Principles of Real-time Object Request Brokers,” BEA, Munich, Germany, September 16th, 1999.
263. “Real-time CORBA – Fact or Fiction,” Siemens CORBA Day, Munich, Germany, September 15th, 1999.
264. “Patterns and Principles of Real-time Object Request Brokers,” Siemens MED, Erlangen, Germany, September 13th, 1999.
265. “Patterns and Principles of Real-time Object Request Brokers,” RT DII COE TWG, Boeing, Seattle, WA August 25th, 1999.
266. “Patterns for Real-time Middleware,” Microsoft, Redmond, WA, August 24th, 1999.
267. “Patterns and Principles of Real-time Object Request Brokers,” Lockheed Martin, Eagan, Minnesota, June 22nd, 1999.
268. “Using the ACE Framework and Patterns to Develop OO Communication Software,” Dreamworks SGK, Glendale, CA, May 5th, 1999.
269. “Why Telecom Reuse has Failed and how to Make it Work for You,” Keynote talk at Nortel Design Forum, Ottawa, CA, April 22nd, 1999.
270. “QoS-enabled Middleware for Monitoring and Controlling High-Speed Networks and Endsystems,” Lucent Bell Labs, Murray Hill, NJ, April 15th, 1999.
271. “Optimization Patterns for High-performance, Real-time Object Request Broker Middleware,” University of California, Irvine, April, 2nd, 1999.
272. “Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems,” Lucent, Columbus, OH, March 18-19 and 25-26, 1999.
273. “Using Design Patterns, Frameworks, and Object-Oriented Communication Systems,” Lucent, Holmdel, NJ, March 1-4, 1999.
274. Chaired a panel on ”Research Directions for Middleware,” NSF PI meeting, Washington, DC, January 24th, 1999.
275. “Principles and Patterns of High-performance Real-time CORBA,” University of Southern California, Los Angeles, CA, December 10th, 1998.

276. "Real-time CORBA for Telecom – Fact or Fiction?," Bellcore, Morristown, NJ, December 1st, 1998.
277. "Design Patterns for Real-time Object Request Brokers," Silicon Valley Patterns Group, San Francisco, November 15, 1998.
278. "Why Reuse has Failed and how to Make it Work for You," Keynote talk at Lucent Software Symposium, October 27th, Murray Hill, NJ, 1998.
279. "Real-time CORBA – Fact or Fiction," Lucent CORBA Forum, Holmdel, NJ, September 29, 1998.
280. "Applying Software Design Patterns and Framework to Telecommunication Applications," Nortel Advanced Software Computing and Technology, Monday, April 6, 1998, Ottawa, Canada.
281. "Patterns and Performance of Real-time Object Request Brokers," University of California, Santa Barbara, February 20, 1998.
282. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of Frankfurt, Germany, February 12th, 1998.
283. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of Illinois, Urbana-Champaign November 12th, 1997.
284. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of Missouri, Kansas City, October 31st, 1997.
285. "Principles and Patterns of High-performance, Real-time Object Request Brokers," IBM T.J. Watson Research, September 15, 1997.
286. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of California, Santa Barbara, August 21st, 1997.
287. "Principles and Patterns of High-performance, Real-time Object Request Brokers," Lucent Technologies, Naperville, IL August 19th, 1997.
288. "Mastering Software Complexity with Reusable Object-Oriented Frameworks, Components, and Design Patterns," 3rd NSA Software Reuse Symposium, August 20th, 1997.
289. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of Utah, Salt Lake City, Utah, August 11th, 1997.
290. "Using the ACE Framework and Design Patterns to Develop Object-Oriented Communication Software," CERN, Switzerland, July 18th, 1997.
291. "Principles and Patterns of High-performance, Real-time Object Request Brokers," CHOOSE symposium, Zurich, Switzerland, July 17th, 1997.
292. Invited keynote speaker for 2^{nd} Component's User Conference, Munich Germany, July 1997.
293. "Principles and Patterns of High-performance, Real-time Object Request Brokers," Lucent Bell Laboratories, Murray Hill, New Jersey, July 9th, 1997.
294. "Using the ACE Framework and Design Patterns to Develop Object-Oriented Communication Software," Lockheed Martin Tactical Systems, Minneapolis, Minnesota, June 26th, 1997.
295. QoS for Distributed Object Computing Middleware – Fact or Fiction?, panel at the Fifth International Workshop on Quality of Service (IWQoS '97), May 22nd, 1997, Columbia University, NYC, USA.
296. "Design Patterns and Frameworks for Developing Object-oriented WWW Clients and Servers," Carleton University, April 11th, 1997.
297. "Principles and Patterns of High-performance, Real-time Object Request Brokers," University of Maryland, College Park, Maryland, April 2nd, 1997.
298. "A High-Performance End system Architecture for Real Time COBRA," SPARTAN Symposium sponsored by US Sprint, Lawrence Kansas, March 18th, 1997.
299. "Experience with CORBA for Communication Systems," Motorola, Chicago, January 24th, 1997.
300. "High-performance CORBA," Bay Area Object Interest Group, Stanford Linear Accelerator Center, California, December 5th, 1996.
301. "Gigabit CORBA – An Architecture for High-performance Distributed Object Computing," Numerical Aerodynamic Simulation group, NASA, Moffett Field, California, December 3rd, 1996.

302. “Towards High-performance, Real-time CORBA,” Distinguished Lecturer at Kansas State University, Manhattan, Kansas, November 7th, 1996.
303. “Gigabit CORBA – An Architecture for High-performance Distributed Object Computing,” University of California, Los Angeles, October 3rd, 1996.
304. “Design Patterns and Frameworks for Object-Oriented Communication Software,” NSA Software Reuse Symposium, August 28th, 1996.
305. “CORBA – the Good, the Bad, and the Ugly,” Lucent Bell-Labs, Naperville, IL, August 22nd, 1996.
306. “Components: the Good, the Bad, and the Ugly,” keynote talk for the 1st Components Users Conference, SIEMENS, Munich, Germany, July 15th, 1996.
307. “Design Patterns for Object-Oriented Communication Software,” IONA Technologies, Ltd, Dublin, Ireland, July 12th, 1996.
308. “OO Design Patterns and Frameworks for Communication Software,” Siemens Corporate Research, Princeton, New Jersey, June 27, 1996.
309. “OO Design Patterns for Concurrent, Parallel, and Distributed Systems,” IBM Centre for Advanced Studies, North York, Ontario, Canada, June 17, 1996.
310. “Distributed Object Computing with CORBA”, Bell Laboratories, Murray Hill, New Jersey, June 11-12th, 1996.
311. “Design Patterns for Object-Oriented Communication Software,” Carleton University, Ottawa, Canada, May 21st, 1996.
312. “Integrating LAN-WAN-Celestial Networks with Design Patterns,” Featured technical session at the Object World East conference, Boston, MA, May 9th, 1996.
313. “Using Design Patterns to Develop Object-Oriented Communication Software Frameworks and Applications,” McMaster’s University, Hamilton, Canada, May 2nd, 1996.
314. “Towards Gigabit CORBA – A High-Performance Architecture for Distributed Object Computing,” University of Nevada, Reno, April 25th, 1996.
315. “Domain Analysis: From Tar Pit Extraction to Object Mania?” Panelist at the 4th International Conference on Software Reuse, Orlando, Florida, April 25^{th}, 1996. (other panelists include Spencer Peterson, SEI CMU, Mark Simos, Organon Motives Inc., Will Tracz, Loral, and Nathan Zalman, BNR Inc).
316. “Concurrent Object-Oriented Network Programming with C++,” Kodak Imaging Technology Center, April 19^{th}, 1996.
317. “Using OO Design Patterns and Frameworks to Develop Object-Oriented Communication Systems,” INRS/NorTel Workshop on Telecommunication Software, Montreal, CA, March 14^{th}, 1996.
318. “Concurrent Object-Oriented Network Programming with ACE and C++,” for Siemens Medical Engineering, Erlangen Germany, February 15^{th}, 1996.
319. “OO Componentware” Panelist at the *OOP ’96 Conference*, SIGS, Munich, Germany, February 13^{st}, 1996. (other panelists included Michael Stal (Siemens AG) and Frank Buschmann (Siemens AG).
320. “Using Design Patterns to Develop High-performance Object-Oriented Communication Software Frameworks,” for the Department of Information Systems, Institute of Computer Science, Johannes Kepler University of Linz, Austria, February 12^{th}, 1996.
321. “The Performance of Object-Oriented Components for High-speed Network Programming,” for the Digital Libraries research group at Stanford University, Palo Alto California, February 2^{nd}, 1996.
322. “Distributed Object Computing with CORBA, ACE, and C++,” for South Western Bell Telephone advanced distributed systems group, St. Louis, MO., January 26^{th}, 1996.
323. “OO Design Patterns for Large-Scale Object-Oriented Communication Software Systems,” AG Communication Systems, Phoenix, Arizona, December $11 - 13^{th}$, 1995.
324. “Experience Using OO Design Patterns to Develop Large-Scale Object-Oriented Communication Software Systems,” Bell Northern Research, 7th Annual Design Forum, Ottawa, Canada, December 6^{th}, 1995.

325. "Using OO Design Patterns to Develop Large-Scale Distributed Systems," Object Technology International, Ottawa, Canada, November 22^{nd}, 1995.
326. "Design Patterns for Concurrent, Parallel, and Distributed Systems," North Dallas Society for Object Technology, September 13^{th}, 1995.
327. "Using Design Patterns for Iridium Communication Services," at Motorola Iridium, Chandler, AZ, June 30^{th}, 1995.
328. "Object Technology and the World-Wide Information Infrastructure," Panelist at ECOOP '95, Aarhus, Denmark, August 9^{th}, 1995.
329. "Measuring the Performance of CORBA over ATM Networks," HP Labs, Palo Alto, CA, June 28^{th}, 1995.
330. "Measuring the Performance of Object-Oriented Components for High-speed Network Programming," The C++ and C SIG user group, New York, New York, June 5^{th}, 1995.
331. "An Overview of Design Patterns for Object-Oriented Network Programming," St. Louis Chapter of the ACM, St. Louis, MO, March 13^{th} 1995.
332. "Design Patterns for Concurrent Object-Oriented Network Programming," Distributed Systems group at Siemens Corporate Research Center, Munich, Germany, March 3^{rd}, 1995.
333. "Patterns: 'Eureka,' 'Deja-Vu,' or 'Just Say No'?" Panelist at the *OOP '95 Conference*, SIGS, Munich, Germany January 31^{st}, 1995. (other panelists included Richard Helm, (DMR), Frank Buschmann (Siemens AG), and Dave Thomas (OTI).
334. "Developing Distributed Applications with the ADAPTIVE Communication Environment," *The 12^{th} Annual Sun Users Group Conference*, SUG, San Francisco, California, June 17^{th}, 1994.
335. "Flexible Configuration of High-performance Distributed Communication Systems," presented at the ETH-Zentrum in the Swiss Federal Institute of Technology, Zurich, Switzerland, May 31^{st}, 1994.
336. "Object Oriented Techniques for Developing Distributed Applications," *Computer Science Department Colloquia*, California State University Northridge, December 7^{th}, 1993.
337. "Hosting the ADAPTIVE System in the *x*-Kernel and System V STREAMS," *The x-Kernel Workshop*, IEEE, Tucson, Arizona, November 10^{th}, 1992.
338. "An Environment for Controlled Experimentation on the Performance Effects of Alternative Transport System Designs and Implementations," IBM T. J. Watson Research Center, Hawthorne, New York, September 10^{th}, 1992.

Colloquia, Seminars, and Tutorials

1. "Programming with Java Lambdas and Streams," O'Reilly Live Training, December 6th, 2021.
2. "Design Patterns in Java," O'Reilly Live Training, November 15th and 16nd, 2021.
3. "Scalable Reactive Programming with Java," O'Reilly Live Training, September 9th, 2021.
4. "Design Patterns in Java," O'Reilly Live Training, September 1st and 2nd, 2021.
5. "Programming with Java Lambdas and Streams," O'Reilly Live Training, July 20th, 2021.
6. "Scalable Reactive Programming with Java," O'Reilly Live Training, May 17th, 2021.
7. "Scalable Reactive Programming with Java," O'Reilly Live Training, January 22nd, 2021.
8. "Programming with Java Lambdas and Streams," O'Reilly Live Training, January 13th, 2021.
9. "Design Patterns in Java," O'Reilly Live Training, November 12th and 13th, 2020.
10. "Design Patterns in Java," O'Reilly Live Training, September 17th and 18th, 2020.
11. "Programming with Java Lambdas and Streams," O'Reilly Live Training, September 14th, 2020.
12. "Core Java Synchronizers," O'Reilly Live Training, August 20th, 2020.
13. "Scalable Reactive Programming with Java," O'Reilly Live Training, August 19th, 2020.
14. "Programming with Java Lambdas and Streams," O'Reilly Live Training, June 1st, 2020.
15. "Design Patterns in Java," O'Reilly Live Training, May 27th and 28th, 2020.

16. “Core Java Synchronizers,” O’Reilly Live Training, May 18th, 2020.
17. “Programming with Java Lambdas and Streams,” O’Reilly Live Training, March 30th, 2020.
18. “Design Patterns in Java,” O’Reilly Live Training, March 23rd and 24th, 2020.
19. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, February 24th, 2020.
20. “Core Java Synchronizers,” O’Reilly Live Training, February 10th, 2020.
21. “Design Patterns in Java,” O’Reilly Live Training, January 29th and 30th, 2020.
22. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, January 22nd, 2020.
23. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, January 22nd, 2020.
24. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, November 27th, 2019.
25. “Reactive Programming with Java 8 CompletableFutures,” O’Reilly Live Training, November 18th, 2019.
26. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, November 6th, 2019.
27. “Design Patterns in Java,” O’Reilly Live Training, November 4th and 5th, 2019.
28. “Design Patterns in Java,” O’Reilly Live Training, September 17th and 18th, 2019.
29. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, September 3rd, 2019.
30. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, August 29th, 2019.
31. “Reactive Programming with Java 8 CompletableFutures,” O’Reilly Live Training, August 15th, 2019.
32. “Design Patterns in Java,” O’Reilly Live Training, July 29th and 30th, 2019.
33. “Reactive Programming with Java 8 CompletableFutures,” O’Reilly Live Training, August 15th, 2019.
34. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, July 2nd, 2019.
35. “Design Patterns in Java,” O’Reilly Live Training, June 13th and 14th, 2019.
36. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, May 16th, 2019.
37. “Reactive Programming with Java 8 CompletableFutures,” O’Reilly Live Training, May 13th, 2019.
38. “Design Patterns in Java,” O’Reilly Live Training, April 17th and 18th, 2019.
39. “Scalable Programming with Java 8 Parallel Streams,” O’Reilly Live Training, March 27th, 2019.
40. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, March 12th, 2019.
41. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, March 5th, 2019.
42. “Design Patterns in Java,” O’Reilly Live Training, February 26th and 27th, 2019.
43. “Reactive Programming with Java 8 CompletableFutures,” O’Reilly Live Training, February 19th, 2019.
44. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, February 5th, 2019.
45. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, January 22nd, 2019.
46. “Design Patterns in Java,” O’Reilly Live Training, January 7th and 8th, 2019.
47. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, December 11th, 2018.
48. “Programming with Java 8 Lambdas and Streams,” O’Reilly Live Training, December 6th, 2018.
49. “Design Patterns in Java,” O’Reilly Live Training, November 13th and 14th, 2018.
50. “Scalable Concurrency with the Java Executor Framework,” O’Reilly Live Training, October 29th, 2018.

51. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, October 16th, 2018.
52. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, October 4th, 2018.
53. "Design Patterns in Java," O'Reilly Live Training, September 18th and 19th, 2018.
54. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, September 4th, 2018.
55. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, August 30th, 2018.
56. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, August 20th, 2018.
57. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, July 25th, 2018.
58. "Design Patterns in Java," O'Reilly Live Training, July 2nd and 3rd, 2018.
59. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, June 26th, 2018.
60. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, June 25th, 2018.
61. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, June 8th, 2018.
62. "Design Patterns in Java," O'Reilly Live Training, May 24th and 25th, 2018.
63. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, April 26th, 2018.
64. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, April 17th, 2018.
65. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, April 13th, 2018.
66. "Design Patterns in Java," O'Reilly Live Training, April 3rd, 2018.
67. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, March 13th, 2018.
68. "Scalable Programming with Java 8 Parallel Streams: Part 2," O'Reilly Live Training, March 7th, 2018.
69. "Scalable Programming with Java 8 Parallel Streams: Part 1," O'Reilly Live Training, March 6th, 2018.
70. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, March 1st, 2018.
71. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, February 13th, 2018.
72. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, February 6th, 2018.
73. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, February 1st, 2018.
74. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, January 12th, 2018.
75. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, January 10th, 2018.
76. "Reactive Programming with Java 8 CompletableFutures," O'Reilly Live Training, January 9th, 2018.
77. "Reactive Programming with Java 8 Completable Futures," O'Reilly Live Training, October 23rd, 2017.
78. "Programming with Java 8 Lambdas and Streams," O'Reilly Live Training, October 19th, 2017.
79. "Scalable Programming with Java 8 Parallel Streams," O'Reilly Live Training, October 17th, 2017.
80. "Java 8 Concurrency," O'Reilly Live Training, September 7-8th, 2017.
81. "Java 8 Concurrency," O'Reilly Live Training, August 30-31st, 2017.
82. "Java 8 Concurrency," O'Reilly Live Training, June 28-29th, 2017.
83. "The C++ Standard Template Library," Qualcomm, San Diego, February 16-19, 2016.
84. "The C++ Standard Template Library," Qualcomm, San Diego, October 13-16, 2015.
85. "The C++ Standard Template Library," Qualcomm, San Diego, October 13-16, 2015.
86. "Pattern-Oriented Java Concurrency," InformIT Webinar, May 14th, 2015.

87. "Pattern-Oriented Concurrent Programming with Java," OOP Conference, Munich, Germany, January 30th, 2015.
88. "Concurrent Programming in Android," OOP Conference, Munich, Germany, January 29th, 2015.
89. "The C++ Standard Template Library," Qualcomm, San Diego, October 14-17, 2014.
90. "The C++ Standard Template Library," Qualcomm, San Diego, August 5-8, 2014.
91. "Pattern-Oriented Software Architecture for Concurrent and Networked Software," July 28-31, 2014.
92. "The C++ Standard Template Library," Qualcomm, San Diego, August 5-8, 2014.
93. "The C++ Standard Template Library," Qualcomm, India, March, 2014.
94. "The C++ Standard Template Library," Qualcomm, San Diego, CA, January 23-34, 2014.
95. "The C++ Standard Template Library," Qualcomm, San Diego, CA, October 16-17th, 2013.
96. "Patterns and Frameworks for Concurrent and Networked Software," 2013 International Summer School on Trends in Computing Tarragona, Spain, July 25-26, 2013.
97. "The C++ Standard Template Library," Qualcomm, San Diego, CA, January 23-24th, 2013.
98. "The C++ Standard Template Library," Qualcomm, San Diego, CA, October 4-5th, 2012.
99. "Embedded Systems Patterns for C Developers," Qualcomm, San Diego, CA, August 28th, September 11th, September 25th, October 9th, October 23rd, and November 6th, 2012.
100. "Embedded Systems Patterns for C Developers," Qualcomm, San Diego, CA, August 14-15th, 2012.
101. "The C++ Standard Template Library," Qualcomm, San Diego, CA, May 15-18th, 2012.
102. "The C++ Standard Template Library," Qualcomm, San Diego, CA, January 25-26th, 2012.
103. "Object-Oriented Software Patterns and Frameworks," Qualcomm, San Diego, CA, October 11-12th, 2011.
104. "The C++ Standard Template Library," Qualcomm, San Diego, CA, May 11-12th, 2011.
105. "The C++ Standard Template Library," Qualcomm, San Diego, CA, January 25-26, 2011.
106. "Pattern-Oriented Software Architecture: A Pattern Language for Concurrent and Networked Software," SPLASH 2010, October 17-21, 2010, Reno, Nevada.
107. "Pattern-Oriented Software Architectures - Patterns and Frameworks for Concurrent and Networked Software," ProObject, Hanover, MD, August 11th, 2010.
108. "Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Embedded Systems," Qualcomm, Bangalore, India, June 21-22, 2010.
109. "Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Embedded Systems," Qualcomm, Hyderabad, India, June 24-25, 2010.
110. "Pattern-Oriented Software Architecture: A Pattern Language for High Quality and Affordable Distributed Computing Systems," IEEE Webinar Series, June 10th, 2010.
111. "The C++ Standard Template Library," Qualcomm, San Diego, CA, May 12-13, 2010.
112. "The C++ Standard Template Library," Qualcomm, San Diego, CA, December 16-17, 2009.
113. "Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing," OOPSLA 2009, Orlando, FL, October, 2009.
114. "The C++ Standard Template Library," Qualcomm, San Diego, CA, September 15-16, 2009.
115. "Networked Embedded Systems Patterns for C Developers," Qualcomm, San Diego, CA, June 11-12, 2009.
116. "Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing," Software Architecture Technology Users' Network (SATURN) workshop May 5, 2009 in Pittsburgh, PA.
117. "The C++ Standard Template Library," Qualcomm, San Diego, CA, January 29-30, 2009.
118. "Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing," IEEE Webinar Series, January 8th, 2009.

119. "Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing," OOPSLA 2008, Nashville, TN, October 20, 2008.
120. "The Data Distribution Service for Real-time Systems," OOPSLA 2008, Nashville, TN, October 19, 2008.
121. "Object-Oriented Patterns for Concurrent and Networked Applications," Qualcomm, San Diego, CA, August 5-6th, 2008.
122. "The C++ Standard Template Library," Qualcomm, San Diego, NJ, July 29-30, 2008.
123. "Object-Oriented Patterns and Frameworks with C++," Qualcomm, San Diego, CA, June 12-13, 2008.
124. "The C++ Standard Template Library," Qualcomm, New Jersey, May 5-6, 2008.
125. "Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing," Software Architecture Technology Users' Network (SATURN) workshop April 28 - May 1, 2008 in Pittsburgh, PA.
126. Developing Distributed Computing Systems with Patterns and Middleware, UCLA Extension, February 19-21, 2008.
127. Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing, OOPSLA 2007, Montreal, CA, October 24, 2007.
128. Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware, Qualcomm, New Jersey, September 27-28, 2007.
129. Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware, Qualcomm, San Diego, CA, August 21-22, 2007.
130. Lightweight CORBA Component Model, 8th OMG Real-time/Embedded CORBA workshop, Washington DC, July 9–12, 2007.
131. Model-Driven Engineering for Distributed Real-time and Embedded Systems, 13th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2007), Bellevue, WA, United States April 3-6, 2007.
132. "Improving Product Reliability and ROI Through Effective Software Reuse," Qualcomm, San Diego, CA, March 27th, 2007.
133. "Developing Distributed Computing Systems with Patterns and Middleware," UCLA Extension, February 21-23, 2007.
134. "POSA: Patterns for Concurrent and Distributed Systems," OOP, Munich, Germany, January 22, 2007.
135. "Meeting the Challenges of Software-Intensive Embedded Systems," OOP, Munich, Germany, January 23, 2007.
136. "Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware," Qualcomm, San Diego, CA, January 10-11, 2007.
137. "Model-Driven Development of Distributed Systems," OOPSLA 2006, Portland, OR, October 22-26, 2006.
138. "Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects," OOPSLA 2006, Portland, OR, October 22-26, 2006.
139. "Model-Driven Engineering of Distributed Systems," MODELS 2006, Genova, Italy, October 1, 2006.
140. "Distributed Real-time and Embedded Systems," Advanced Institute of Information Technology, Seoul, Korea, August 7-11 2006.
141. "Lightweight CORBA Component Model," 7th OMG Real-time/Embedded CORBA workshop, Washington DC, July 10–13, 2006.
142. "How to Use ACE Effectively," Trion World Network, Austin, TX, June 19-21, 2006.
143. "Improving Product Reliability and ROI Through Effective Software Reuse," Qualcomm, San Diego, CA, June 15, 2006.

144. "Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware," Qualcomm, San Diego, CA, June 13-14, 2006.
145. "Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware," Qualcomm, San Diego, CA, Feb 9-10, 2006.
146. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, University of California, Los Angeles Extension, January 18-20st, 2006."
147. "Model Driven Development of Distributed Real-time and Embedded Systems," at the OOP conference, January 17, 2006, Munich, Germany.
148. "Pattern-Oriented Software Architecture," at the OOP conference, January 16, 2006, Munich, Germany.
149. "Model Driven Development: State of the Art," at the OOP conference, January 16, 2006, Munich, Germany.
150. "Concurrent C++ Network Programming with Patterns and Frameworks," C++ Connections: 20 Years of C++ conference, November 11, 2005, Mandalay Bay, Las Vegas, NV.
151. "Pattern-Oriented Software Architecture: Patterns for Concurrent and Distributed Systems," OOPSLA 2005, San Diego, October 17th, 2005.
152. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," BAE Systems, Greenlawn, New York, August 25, September 2-3.
153. "Lightweight CORBA Component Model," 6th OMG Real-time/Embedded CORBA workshop, Washington DC, July 11–14, 2005.
154. "Model Driven Development for Distributed Real-time and Embedded Systems," OMG Information Days: MDA - Frankfurt, Germany, June 9th, 2005
155. "Model Driven Development for Distributed Real-time and Embedded Systems," OMG Information Days: MDA - Munich, Germany, June 7th, 2005.
156. "Model Driven Development for Distributed Real-time and Embedded Systems," OMG Information Days: MDA - Zurich, Switzerland, June 1st, 2005.
157. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," BAE Systems, Wayne, New Jersey, May 13, 16, 19, 23, 27, 2005.
158. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," BAE Systems, Wayne, New Jersey, February 18th, February 22nd, March 1, 8, and 15 2005.
159. "Pattern-Oriented Software Architectures for Distributed Systems" the OOP conference, January 28, 2005, Munich, Germany.
160. "Research on Model Driven Development of Distributed Real-time and Embedded Systems," the OOP conference, January 26, 2005, Munich, Germany.
161. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, January 19-21st, 2005.
162. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, BAE Systems, Wayne, New Jersey, October 29, November 1, 8, 15, 22, 2004.
163. "Pattern-Oriented Software Architectures for Distributed Systems," OOPSLA 2004, Vancouver, British Columbia, October 25th, 2004.
164. "Notes on the Forgotten Craft of Software Architecture", OOPSLA 2004, Vancouver, British Columbia, October 25th, 2004.
165. "Model Driven Architecture with QoS-enabled component middleware," MDE for Embedded Systems, Brest, France, September 10th 2004.
166. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Qualcomm, San Diego, CA, Jan 7-6, 2005.
167. "Object-Oriented Design and Programming with Patterns, Frameworks, and Middleware," Qualcomm, San Diego, CA, Jan 9-10, 2005.

168. "Using the Lightweight CORBA Component Model to Develop Distributed Real-time and Embedded Applications," OMG Workshop on Distributed Object Computing for Real-time and Embedded Systems, July 12th, 2004, Reston, VA.
169. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, July 7-9th, 2004.
170. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, University of California, Los Angeles Extension, January 21st-23rd, 2004.
171. Patterns and Frameworks for Concurrent Distributed Systems, SIGS OOP Conference, Munich, Germany, January 19th, 2004.
172. Middleware for Distributed Real-time and Embedded Systems, SIGS OOP Conference, Munich, Germany, January 19th, 2004.
173. "Pattern-Oriented Software Architectures for Networked and Concurrent Applications," OOPSLA 2003, Anaheim, CA, October 27, 2003.
174. The JAOO 2003 conference, September 22-26, Aarhus, Denmark.
175. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, July 9-11th, 2003.
176. "Patterns, Frameworks, and Middleware: Their Synergistic Relationship," Frontiers of Software Practice, International Conference on Software Engineering, Portland, Oregon, May 7, 2003.
177. "Pattern-Oriented Distributed Systems Architecture," International Conference on Software Engineering, Portland, Oregon, May 5, 2003.
178. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, January 22nd-24th, 2003.
179. "Patterns and Application Experiences for Real-time Object Request Brokers," OOPSLA 2002, Seattle, Washington, November, 2002.
180. "Pattern-Oriented Software Architectures for Networked and Concurrent Applications," OOPSLA 2002, Seattle, Washington, November, 2002.
181. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, Raytheon, St. Petersburg, FL, September 3–5, 2003.
182. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, University of California, Los Angeles Extension, July 22nd-24th, 2002.
183. "Policies and Patterns for High-performance, Real-time Object Request Brokers," Mercury Computer Systems, Tysons Corner, VA, November Feb 7, 2002.
184. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, University of California, Los Angeles Extension, January 23rd-25th, 2002.
185. "Policies and Patterns for High-performance, Real-time Object Request Brokers," Raytheon, Rosslyn, VA, November 12th, 2001.
186. "Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects," OOPSLA 2001, October 15th, 2000, Minneapolis, Minnesota.
187. "Policies and Patterns for High-performance, Real-time Object Request Brokers," International Symposium on Distributed Object Applications (DOA), Rome, September 17-20, 2001.
188. "Policies and Patterns for QoS-enabled Middleware," The JAOO 2001 conference, September 10-14, Aarhus, Denmark.
189. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, July 23rd-25th, 2001.
190. "Policies and Patterns for High-performance, Real-time Object Request Brokers," OMG Second Workshop on Real-time and Embedded Distributed Object Computing on June 4-7, 2001 in Herndon, VA, USA.
191. "Design Patterns for Understanding Middleware and Component Infrastructures," 6th USENIX Conference on Object-Oriented Technologies and Systems, January 29, 2001, San Antonio, TX.

192. "Principles and Patterns of High-performance, Real-time Object Request Brokers," OOP conference, Munich, Germany, January 23, 2001.
193. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, January 3-5, 2001.
194. "Patterns for Concurrent and Distributed Objects," OOPSLA 2000, October 16th, 2000, Minneapolis, Minnesota.
195. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Berkeley Extension, May 24-26, 2000.
196. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Jet Propulsion Laboratory, Pasadena, CA, April, 2000.
197. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Los Angeles Extension, March 27-31, 2000.
198. "Optimizing Middleware to Support High-Performance Real-time Distributed and Embedded Systems," OOP conference, Munich, Germany, January 27, 2000.
199. "Effective Architectures for DOC," OOP conference, Munich, Germany, January 24, 2000.
200. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California, Berkeley Extension, December 13-15, 1999.
201. "Middleware Techniques and Optimizations for Real-time Embedded Systems," 12th International Symposium On System Synthesis, IEEE, San Jose, CA, USA November, 11, 1999
202. "Patterns and Principles of Real-time Object Request Brokers," OOPSLA 1999, ACM, Denver, Colorado, November 1-5, 1999.
203. "Using Design Patterns, Frameworks and CORBA to Reduce the Complexity of Developing Reusable Large-Scale Object-Oriented Concurrent Communication Components and Systems," Fifth IEEE International Conference on Engineering of Complex Computer Systems, Las Vegas, Nevada, October 18-21, 1999
204. "Distributed Technologies," Motorola, Schaumburg, IL, August 10-12, 1999.
205. "Patterns and Principles of Real-time Object Request Brokers," the 3rd Components Users Conference, SIEMENS, Munich, Germany, July 12th, 1999.
206. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Lucent, Naperville, IL, June 23-24 and June 30 - July 1st, 1999.
207. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Motorola Software Symposium, Ft. Lauderdale, Florida, June 21st, 1999.
208. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California Los Angeles Extension, June 2-4, 1999.
209. "Concurrent Object-Oriented Network Programming and Distributed Object Computing," University of California Berkeley Extension, May 19-21, 1999.
210. "Patterns and Principles of Real-time Object Request Brokers," 5th USENIX Conference on Object-Oriented Technologies and Systems, May 4, 1999, San Diego, CA.
211. "Real-time CORBA for Telecom – Fact or Fiction?" Nortel Design Forum, Ottawa, CA, April 22, 1999.
212. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Lucent, Columbus, OH, March 18-19 and 25-26, 1999.
213. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Lucent, Holmdel, NJ, March 1-4, 1999.
214. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Lucent/Octel, Milpitas, CA, December 14-16, 1998.
215. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California Los Angeles Extension, December 8-10, 1998.
216. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Motorola, Schaumburg, IL, December 2-4, 1998.

217. "Concurrent Object-Oriented Network Programming and Distributed Object Computing," University of California Berkeley Extension, November 16-18, 1998.
218. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Software," OOPSLA 1998, October 19th, 1998, Vancouver, British Columbia.
219. "High-Performance CORBA," Lucent CORBA Forum, Holmdel, NJ, September 29, 1998.
220. "Writing Efficient Multi-Thread CORBA Applications," the 3rd Components Users Conference, SIEMENS, Munich, Germany, July 10, 1998.
221. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Software," UCLA extension course, Milan, Italy, June 29 - July 1, 1998.
222. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," Lucent, Naperville, IL, June 8-11, 1998.
223. "Patterns and Performance of Real-time Object Request Brokers," Fourth IEEE Real-Time Technology and Applications Symposium (RTAS), Denver, Colorado, June 5, 1998.
224. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California Los Angeles Extension, June 1-3, 1998.
225. "Patterns and Principles of Real-time Object Request Brokers," NSA, Ft. Meade, MD, March 22, 1998.
226. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, Crosskeys, Ottawa Canada, March 19-21, 1998.
227. "Concurrent Object-Oriented Network Programming and Distributed Object Computing," University of California Berkeley Extension, March 4-6, 1998.
228. "Building Distributed Communication Software with CORBA," the Motorola Systems Symposium, February, 1998, Austin, Texas, USA.
229. "Introduction to Distributed Objects with CORBA," SIGS OOP '98, February 9-13, 1998, Munich, Germany.
230. "Design Patterns for Developing and Using CORBA Object Request Brokers," SIGS OOP '98, February 9-13, 1998, Munich, Germany.
231. Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems, Lucent Technologies, Whippany, NJ, January 5-6, 1998.
232. "Using Design Patterns, Frameworks, and CORBA to Develop Object-Oriented Communication Systems," University of California Los Angeles Extension, December 10-12, 1997.
233. "Concurrent Object-Oriented Network Programming and Distributed Object Computing," University of California Berkeley Extension, December 10-12, 1997.
234. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Systems," Motorola Cellular Infrastructure Group, Arlington Heights, Illinois, December 1 - 3, 1997.
235. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Systems," TOOLS Pacific '97, Melbourne, Australia November 24 - 27, 1997.
236. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Systems" for the IEEE GLOBECOM '97 conference, Phoenix, AZ, November 4-8, 1997.
237. "High-performance Distributed Object Computing with CORBA," IEEE International Conference on Network Protocols, Atlanta, GA, October 28th, 1997.
238. "Using Design Patterns and Frameworks to Develop Object-Oriented Communication Systems," OOPSLA 1997, ACM, Atlanta, GA, October 6-7th, 1997.
239. "Using Design Patterns and Frameworks to Develop Object-oriented Communication Systems," 24th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Asia '97). Beijing, China, September 22, 1997.
240. "Principles and Patterns of Distributed Object Computing Systems," for the ACM Principles of Distributed Computing Conference (PODC), Santa Barbara, CA, August 21st, 1997.
241. "Distributed Object Computing with CORBA and ACE," Alta Software, Jacksonville, FL, June 4-5th, 1997.

242. "Distributed Object Computing with CORBA", Object Expo, NY, NY, June 2nd, 1997.
243. "Concurrent Object-Oriented Network Programming and Distributed Object Computing," University of California Berkeley Extension, May 28-30, 1997.
244. "Patterns and Principles of Real-time Object Request Brokers," National Security Agency, Ft. Meade, MD, May 13th, 1997.
245. "Building Distributed Communication Software with CORBA," the Motorola Systems Symposium, March, 1997, Chandler, AZ, USA.
246. "Evaluating Concurrency Models for CORBA Servers," the 2nd Components Users Conference, SIEMENS, Munich, Germany, July 14th, 1997.
247. "Design Patterns for Evolving System Software Components from UNIX to Windows NT," the 2st Components Users Conference, SIEMENS, Munich, Germany, July 14th, 1997.
248. "Techniques and Patterns for Distributed Object Computing with CORBA and C++," University of California Berkeley Extension, December 4-6, 1996.
249. "Design Patterns for Concurrent Object-Oriented Programming with ACE and C++," C++ World, Dallas, TX, November 11th, 1996.
250. "Implementing Concurrent CORBA Applications with Multi-Threaded Orbix and ACE," C++ World, Dallas, TX, November 12th, 1996.
251. "Why Reuse has Failed, and How You Can Make it Work for You," Berne Technology Forum 1996, Berne, Switzerland, October 18, 1996.
252. "Introduction to Distributed Object Programming with CORBA," the Local Computer Networks '96 conference, IEEE, Minneapolis, Minnesota, October 13, 1996.
253. "Object-Oriented Design Patterns for Concurrent, Parallel, and Distributed Systems," the OOPSLA 1996 conference, ACM, San Jose, California, October, 1996.
254. "OO Design Patterns Network Programming in C++," Object Expo Europe, London, England, September 23rd, 1996.
255. "Effective Multithreaded CORBA Programming," Object Expo Europe, London, England, September 24th, 1996.
256. "Workshop on Object Oriented Technologies," Mitsubishi, July 22nd to July 26th, 1996, Kobe, Japan.
257. "Evaluating Concurrency Models for CORBA Servers," the 1st Components Users Conference, SIEMENS, Munich, Germany, July 15th, 1996.
258. "Design Patterns for Evolving System Software Components from UNIX to Windows NT," the 1st Components Users Conference, SIEMENS, Munich, Germany, July 15th, 1996.
259. "OO Design Patterns for Concurrent, Parallel, and Distributed Systems," the 2^{nd} *Conference on Object-Oriented Technology*, USENIX, Toronto, Canada, June 17, 1996.
260. "OO Design Patterns for Concurrent, Parallel, and Distributed Systems," the 3^{rd} *Conference on Object-Oriented Technology*, USENIX, Portland, Oregon, June 16th, 1996.
261. "OO Design Patterns for Network Programming in C++," the *Object Expo '96 Conference*, SIGS, Sydney, Australia, June 3^{rd}, 1996.
262. "Effective Multi-threaded CORBA Programming Programming," the *Object Expo '96 Conference*, SIGS, Sydney, Australia, June 5^{th}, 1996.
263. "Concurrent Object-oriented Network Programming with C++," University Of California Berkeley Extension, Berkeley, California, May $22^{nd} - -24^{th}$, 1996.
264. "Experience Developing Reusable Software Using Object-Oriented Design Patterns and Frameworks," the 4^{th} *International Conference on Software Reuse*, Orlando, Florida, USA April 23-26, 1996.
265. "Techniques for Object-Oriented Network Programming," the *OOP Conference*, SIGS, Munich, Germany, Feb 14th, 1996.
266. "Using Object-Oriented Design Patterns to Develop Large-Scale Distributed Systems," the *OOP Conference*, SIGS, Munich, Germany, Feb 13^{th}, 1996.

267. "Concurrent Object-oriented Network Programming with C++," University Of California Berkeley Extension, Berkeley, California, November 30th-December 1st, 1995.
268. "Using Object-Oriented Design Patterns to Develop Large-Scale Distributed Systems," the 4^{th} *C++ World Conference*, SIGS, Chicago, Illinois, October 31st, 1995.
269. "Techniques for Object-Oriented Network Programming," the 4^{th} *C++ World Conference*, SIGS, Chicago, Illinois, October 31st, 1995.
270. "Experience using OO Design Patterns to Develop Large-scale Distributed Communication Systems," *OOPSLA 1995 Conference* in Austin, Texas, October 1995.
271. "Concurrent Object-oriented Network Programming with C++," the 9^{th} *European Conference on Object-Oriented Programming (ECOOP)*, Aarhus, Denmark, August, 1995.
272. "Concurrent Object-Oriented Network Programming with C++," the 1^{st} *Conference on Object-Oriented Technology*, USENIX, Monterey, California, June 23, 1995.
273. "Design Patterns for Concurrent and Distributed Systems," the *Object Expo '95 Conference*, SIGS, New York, NY, June 5^{th} 1995.
274. "Object Oriented Network Programming," the *Object Expo '95 Conference*, SIGS, New York, NY, June 5^{th}, 1995.
275. "Software Construction with Active Objects in C++," the *OOP '95 Conference*, SIGS, Munich, Germany January 31, 1995.
276. "Object-Oriented Concurrent Programming with C++," the *OOP '95 Conference*, SIGS, Munich, Germany January 31, 1995.
277. "Concurrent Object-Oriented Programming," the *Winter USENIX Conference*, USENIX, New Orleans, Louisiana, January, 1995.
278. "Object-Oriented Network Programming with C++," the 3^{rd} *C++ World Conference*, SIGS, Austin, Texas, November 14, 1994.
279. "Object-Oriented Techniques for Dynamically Configuring Concurrent Distributed Applications," the 9^{th} *OOPSLA 1994*, ACM, Portland, Oregon, October 23, 1994.
280. "Object-Oriented Network Programming," the 6^{th} *C++ Conference*, USENIX, Cambridge, Massachusetts, April 11, 1994.
281. "Object-Oriented Techniques for Developing Extensible Network Servers," the 2^{nd} *C++ World Conference*, SIGS, Dallas, Texas, October 19, 1993.

Professional Activities

Editorial Activities

1. Guest co-editor for a special issue of the Springer Journal Annals of Telecommunications on "Middleware for Internet distribution in the context of Cloud Computing and the Internet of Things," 2016, with Gordon Blair and Chantal Taconet.
2. Guest co-editor of the Proceedings of the IEEE special issue on Applications of Augmented Reality Environments, 2014.
3. Guest co-editor of the International Journal of Network Protocols and Algorithms (NPA) Special Issue on Data Dissemination for Large scale Complex Critical Infrastructures, 2010.
4. Wrote the foreword to the book *Patterns of Parallel Software Design* by Jorge Luis Ortega Arjona, Wiley, 2010.
5. Editorial board member of the Springer Journal of Internet Services and Applications (JISA).
6. Editorial board member of the Transactions on Pattern Languages of Programming (TPLoP) published by Springer-Verlag.
7. Wrote the foreword to the book *Practical Software Factories in .NET*, by Gunther Lenz and Christoph Wienands, Apress, 2006.
8. Guest editor of the IEEE Computer Special Issue on Model Driven Development, February 2006.

9. Guest co-editor of IEEE Network special issue on "Middleware Technologies for Future Communication Networks," February 2004 (co-editors with Gordon Blair and Andrew Campbell).
10. Editorial board member of the Springer Journal of Aspect-Oriented Software Development.
11. Wrote the foreword to the book *Fundamentals of Distributed Object Systems: The CORBA Perspective*, by Zahir Tari and Omran Bukhres, Wiley and Sons, 2001.
12. Wrote the foreword to the book *Design Patterns in Communication Software*, edited by Linda Rising, Cambridge University Press, 2000.
13. Guest editor of the Special Issue on Components and Patterns for *The Journal of Theory and Practice of Object Systems*, Wiley & Sons, to appear 2002.
14. Invited editorial on "Trends in Distributed Object Computing" for the special issue on Distributed Object-Oriented Systems appearing in the Parallel and Distributed Computing Practices journal, edited by Maria Cobb and Kevine Shaw, Vol. 3, No. 1, March 2000.
15. Co-editor of "Building Application Frameworks: Object-Oriented Foundations of Framework Design," John Wiley & Sons, 1999 (co-editors are Mohamed Fayad and Ralph Johnson), ISBN 0-471-24875-4.
16. Co-editor of "Implementing Application Frameworks: Object-Oriented Frameworks at Work," John Wiley & Sons, 1999 (co-editors are Mohamed Fayad and Ralph Johnson), ISBN 0-471-25201-8.
17. Guest editor of the Special Issue on OO Application Frameworks for the Communications of the ACM, (co-editor Mohamed Fayad), ACM, October, 1997.
18. Guest editor of the special issue on Distributed Object Computing for USENIX Computing Systems Journal, November/December, 1996.
19. Guest editor of a feature topic on Distributed Object Computing for IEEE Communications Magazine, February, 1997.
20. Wrote the foreward for Dr. Nayeem Islam's book on *Distributed Objects: Methodologies for Customizing Operating Systems* (IEEE Computer Society Press, 1996).
21. Guest editor of the Special Issue on Patterns and Pattern Languages for Communications of the ACM, (co-editors Ralph Johnson and Mohamed Fayad), ACM, October, 1996.
22. Co-editor of a book entitled "Pattern Languages of Program Design," Addison-Wesley, 1995 (co-editor is Jim Coplien, Bell Labs).
23. Editor of the Patterns++ section of the C++ Report Magazine, April 1997 - March 1998.
24. Editor-in-chief of the C++ Report Magazine, January 1996 - February 1997.
25. Editorial board member of the IEEE Computer Society - Computer Science & Engineering Practice Board.

Program Chairmenships and Conference Organization

1. Chair of the DoD Organic Software Infrastructure Workshop, Arlington VA, August 13th, 2018.
2. General Chair of the Software Product Line Conference, Nashville TN, July/August, 2015.
3. Program Chair of the Interoperable Open Architecture 2013 conference, September 10-11, 2013, Washington, DC.
4. Program Chair of the NSF Workshop on Computing Clouds for Cyber-Physical Systems, March 15th, 2013, Ballston, VA.
5. Program Chair of the Interoperable Open Architecture 2012 conference, October 29-31, 2012, London, UK.
6. Program co-chair for the 1st International Symposium on Secure Virtual Infrastructures (DOA-SVI'11), 17-19 Oct 2011, Crete, Greece.
7. Program co-chair for the COMmunication System softWAre and middleware (Comsware) conference, Helsinki, Finland, August 2010.
8. Doctoral symposium chair for OOPSLA 2009, Orlando Floria, October 25-29, 2009.
9. General co-chair for the 3rd ACM International Conference on Distributed Event-Based Systems (DEBS 2009), July 6-9, 2009 - Nashville, TN, USA.

10. Member of the ISORC 2009 advisory and publicity committee for ISORC 2009, March 17-20, 2009, Toyko, Japan.
11. Area Coordinator for the Integrating Systems of Systems using Services topic at the 6th International Conference on Service Oriented Computing, Sydney (Australia), December 1st - 5th, 2008.
12. Member of the Advisory and Publicity Committee for ISORC 2008, Orlando, Florida, May 5 -7, 2008.
13. Co-chair of the Middleware for Network Eccentric and Mobile Applications (MiNEMA.08) Workshop co-located with ACM EuroSys Conference, March 31 - April 1, 2008, Glasgow, Scotland.
14. General chair of the ACM/IEEE 10th International Conference on Model Driven Engineering Languages and Systems (MODELS 2007), Nashville TN, September 30-October 5, 2007.
15. Area co-coordinator for the Quality of Service research track at the The Fifth International Conference on Service-Oriented Computing, September 17-20, 2007, Vienna, Austria.
16. Program co-chair of the NSF workshop on New Research Directions in Composition and Systems Technology for High Confidence Cyber Physical Systems, July 9, 2007.
17. Program co-chair for the Science of Design Principal Investigators workshop, February 28 to March 2, 2007.
18. Program co-coordinator for SOA Runtime area of the 4th International Conference on Service Oriented Computing Chicago, USA, December 4-7, 2006.
19. Program co-chair of the NSF/NCO Workshop on High-Confidence Software Platforms for Cyber-Physical Systems (HCSP-CPS) Workshop systems, November 30th to December 1st, 2006, Alexandria, VA.
20. Panels chair for the MoDELS 2006 conference, Genova Italy, Oct. 2-6, 2006.
21. Program Co-Chair of the Generative Programming and Component Engineering (GPCE) Conference, Portland, OR, October 2006 (collocated with OOPSLA '06).
22. Program Chair of the NSF/NCO Workshop on New Research Directions in High Confidence Software Infrastructure for Distributed Real-time and Embedded (DRE) systems, July 10th, 2006, Fairfax VA.
23. Program Co-Chair of the NSF/NCO High Confidence Medical Device Software and Systems (HCMDSS) Workshop, May 2005, University of Pennsylvania, Philadelphia, Pennsylvania.
24. Track Vice Chair for Real-time Middleware and Software Engineering for the Real-time Systems Symposium, Lisbon, Portugal, December, 2004.
25. Program Co-chair for the NSF/NCO Planning Meeting for the High Confidence Medical Device Software and Systems (HCMDSS) Workshop, November 16-17, 2004, Arlington, VA.
26. Program chair for 19th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOSPLA), October 24-28, 2004, Vancouver, British Columbia, Canada.
27. General co-chair of the IEEE Real-Time and Embedded Technology and Applications Symposium, May 25 – 28, 2004, Toronto, Canada.
28. Program chair of the CCM Workshop, December 10th, 2003, Nashville, TN.
29. General co-chair for the 5th International Symposium on Distributed Objects and Applications, November 3–7 2003, Catania, Sicily.
30. Program co-chair of the 3rd TAO Workshop, July 18, 2002, Arlington, VA.
31. Program co-chair for Middleware 2003, 4th IFIP/ACM/USENIX International Conference on Distributed Systems Platforms, June 16-20, 2003, Rio de Janeiro, Brazil.
32. Program co-chair for the 9th IEEE Real-time/Embedded Technology and Applications Symposium (RTAS), May 27-30, 2003, Washington, DC.
33. Area vice-chair and session chair for Middleware at the 23rd IEEE International Conference on Distributed Computing Systems (ICDCS), May 19-22nd, 2003, Providence, RI.
34. Program co-chair of the IEEE Workshop on LargeScale Real-Time and Embedded Systems, December 2, 2002, Austin, TX.

35. Program co-chair for the 4th International Symposium on Distributed Objects and Applications, October 28–November 1, 2002, Irvine, CA.
36. Co-organizer of the cross-agency Software Design and Productivity Coordinating Group Workshop on New Visions for Software Design and Productivity: Research and Applications, December 13-14, Nashville, TN.
37. Program co-chair for the 3rd International Symposium on Distributed Objects and Applications, September 18-20, 2001, Rome, Italy.
38. Co-organizer of the cross-agency Workshop on New Visions for Software Design and Productivity, April 18-19, 2000, Ballston, VA.
39. Area vice-chair and session chair for Middleware at the IEEE International Conference on Distributed Computing Systems, April 16-19, Phoenix, AZ, 2001.
40. Tutorial chair for the 6th USENIX Conference on Object-Oriented Technologies and Systems, January 27 - February 3, 2001, San Antonio, TX.
41. Co-chair of the OMG Workshop on Real-time and Embedded CORBA, in Reston, VA, July 24-27, 2000.
42. General chair of the IFIP/ACM International Conference Middleware 2000 in New York, April, 2000.
43. Tutorial chair for the 5^{th} USENIX Conference on Object-Oriented Technologies and Systems, May 3-7, 1999, San Diego, CA.
44. Treasurer for the Fourth International Workshop on Object-oriented Real-time Dependable Systems (WORDS'99) January 27-29, 1999, Radisson Hotel, Santa Barbara, California, USA.
45. Tutorial chair for the 4^{th} USENIX Conference on Object-Oriented Technologies and Systems, April 27-30, 1998, Santa Fe, New Mexico.
46. Co-chair of the mini-track on Engineering Client-Server Systems for the HICSS-31 conference, the Big Island of Hawaii - January 6-9, 1998.
47. Tutorial chair for the 3^{rd} USENIX Conference on Object-Oriented Technologies and Systems, Portland, OR, June 1997.
48. Publicity chair for the 5^{th} IEEE International Workshop on Object-Orientation in Operating Systems, IEEE TCOS and USENIX, Seattle, Washington, October 27-28, 1996.
49. Program chair for 3^{rd} conference on Programming Languages of Programming, Allerton, IL, USA, September, 1996.
50. Program chair for the 2^{nd} USENIX Conference on Object-Oriented Technologies, June 1996.

Professional Service and Advisory Positions

1. Member of the Fraunhofer Advisory Board for the University of Maryland, College Park.
2. Member of the steering committee for the Software Product-Line Conference series.
3. Member of the Future Airborne Capabilities Environment (FACE) Advisory Board.
4. Vice-Chair of the Cyber Situation Awareness study for the Air Force Scientific Advisory Board.
5. Member of the Joint Tactical Radio System (JTRS) Tiger Team in support of the Assistant Secretary of the Army, Acquisition, Logistics, and Technology.
6. Member of the Air Force Scientific Advisory Board.
7. Member of the advisory board for the NSF-sponsored Repository for Model-Driven Development (ReMoDD) project at Colorado State University.
8. Member of the National Academics Committee on Advancing Software-Intensive Systems Producibility, chaired by Bill Scherlis from Carnegie Mellon University (CMU).
9. Member of the Engineering and Methods Technical Advisory Group (TAG) for the Software Engineering Institute at Carnegie Mellon University (CMU) from 2006 to 2009.
10. Member of the Ultra-Large-Scale (ULS) Systems study commissioned by the US Army and conducted at the Software Engineering Institute at Carnegie Mellon University (CMU).

11. Member of the Joshua group, which is an advisory board for the Air Force Research Lab (AFRL) in Rome, NY.
12. Member of the steering committee for the Distributed Objects and Applications conference series.
13. Member of the steering committee for the ACM/USENIX/IFIP Middleware conference series.
14. Member of the steering committee for EMSOFT 2002: Second Workshop on Embedded Software, Grenoble, France, October, 7–9th, 2002.
15. Member of the steering committee for EMSOFT 2001: First Workshop on Embedded Software, Lake Tahoe, California, October, 8th–10th, 2001.
16. Member of the Board of Directors for the Embedded Systems Consortium for Hybrid and Embedded Research (ESCHER).
17. Member of the NASA/JPL Mars Science Laboratory Mission Concept Review Board.
18. Chair of the subcommittee on Embedded and Hybrid Systems program for the National Science Foundation's 2003 Committee of Visitors in the Computer and Communications Research (C-CR) Division.
19. Co-chair of the Software Design and Productivity (SDP) Coordinating Group of the Federal government's multi-agency Information Technology Research and Development (IT R&D) Program, the collaborative IT research effort of the major Federal science and technology agencies. The SDP Coordinating Group formulates the multi-agency research agenda in fundamental software design.
20. One of the three founding members of the Scientific Advisory Board for the *International Symposium of Distributed Objects and Applications* conference series.
21. Member of the advisory board for Entera, which provides Internet content delivery systems based on ACE.
22. Invited to participate in the OO Working Group of the "Strategic Directions in Computing Research" workshop sponsored by ACM at MIT in June 1996.

Technical Program Committees

1. The 3rd IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS 2022) held virtually from 19th to 23rd September 2022.
2. The 16th ACM International Conference on Distributed and Event-Based Systems, June 27 to July 1, 2022, Copenaghen, Denmark.
3. 8th International Workshop on Middleware and Applications for the Internet of Things (M4IoT), held in December 2021 in conjunction with the ACM/IFIP International Middleware Conference.
4. Middleware 2021 Doctoral Symposium, Dec. 6-10, 2021 in Quebec Canada.
5. The 2nd IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS 2021), September 27 to October 1, 2021, Washington DC, USA.
6. "Web of Things, Ubiquitous and Mobile Computing" Track for the Web Conference 2021, Ljubljana, Slovenia, from April 19-23, 2021.
7. 7th International Workshop on Middleware and Applications for the Internet of Things (M4IoT), December 2020 in conjunction with the ACM/IFIP International Middleware Conference.
8. 14th ACM International Conference on Distributed and Event-based Systems, July 13 to July 17, 2020, in Montreal, Quebec, Canada.
9. The Web Conference 2020: Web of Things, Ubiquitous, and Mobile Computing Track, April 20-24th, 2020, Taipei, Taiwan.
10. 6th Middleware for Context-Aware Applications in the IoT (M4IOT) workshop collocated with the ACM/IFIP/USENIX Middleware 2019 Conference, UC Davis, California, USA, December 9-13th 2019.
11. IEEE Workshop on IoT Big Data and Blockchain, at the 2019 IEEE International Conference on Big Data (IEEE Big Data 2019), December 9-12, 2019, Los Angeles, CA, USA.
12. The Second International Workshop on Blockchain Dependability, in conjunction with SRDS2019, Lyon, France, October 1, 2019.

13. The 13th ACM International Conference on Distributed and Event-based Systems, 4th-28th June, 2019, Darmstadt, Germany.
14. The "Web of Things, Ubiquitous, and Mobile Computing" track of The Web Conference 2019, San Francisco, CA, USA, May 13–17, 2019.
15. 17th Workshop on Adaptive and Reflexive Middleware (ARM), collocated with ACM/IFIP/Usenix Middleware 2018, December 10-14th, 2018, Rennes, France.
16. 25th International Conference on Pattern Languages of Programs (PLoP 2018), October 23 – 26th, Portland, OR, USA.
17. First International Workshop on Blockchain Dependability (WBD2018), held in conjunction with the 14th European Dependable Computing Conference, 10-14 September 2018, Iasi, Romania.
18. Workshop on Designing Resilient Intelligent Systems for Testability and Reliability, April 30 – May 4, 2018 in Seattle, USA (co-located with ICSA 2018).
19. 15th IEEE International Conference on Autonomic Computing (ICAC 2018), Sept 3-7, 2018, Trento, Italy.
20. International Conference on Information Society and Smart Cities (ISC 2018), Oxford city, United Kingdom 06-07 June, 2018.
21. 16th Workshop on Adaptive and Reflective Middleware workshop collocated with the ACM/IFIP/USENIX Middleware 2017 Conference, Las Vegas, Nevada, Dec 11-15, 2017.
22. 4th Middleware for Context-Aware Applications in the IoT (M4IOT) workshop collocated with the ACM/IFIP/USENIX Middleware 2017 Conference, Las Vegas, Nevada, Dec 11-15, 2017.
23. 10th International Workshop on Dynamic Software Product Lines - Adaptive Systems through Runtime Variability (DSPL '17), Sept 25-29, 2017, Sevilla, Spain.
24. 11th ACM International Conference on Distributed and Event-Based Systems (DEBS 2017), June 19 - 23, 2017, Barcelona, Spain.
25. 3rd Middleware for Context-Aware Applications in the IoT (M4IOT) workshop collocated with the ACM/IFIP/USENIX Middleware 2016 Conference, December 12-16, 2016 - Trento, Italy.
26. 7th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation, October 5th – 14th, 2016, Corfu, Greece.
27. 10th ACM International Conference on Distributed and Event-based Systems, June 20 to June 24, 2016 in Irvine, CA.
28. First International Workshop on Science of Smart City Operations and Platforms Engineering (SCOPE), April 11, 2016, Vienna, Austria (Co-located with CPS Week).
29. 9th Dynamic Software Product Lines (DSLP) 2015 (held as part of SASO 2015) at MIT on September 21, 2015.
30. 13th International Conference on Advances in Mobile Computing and Multimedia (MoMM2015), Brussels, Belgium from 10-12 December 2015.
31. 13th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC 2015, track on Cyber Physical Systems, Porto Portugal, October 21-23, 2015.
32. 35th IEEE International Conference on Distributed Computing Systems (ICDCS), June29 - July 2, 2015 in Columbus, Ohio, USA.
33. Fourth International Conference on Emerging Applications of Information Technology (EAIT) at Indian Statistical Institute, Kolkata, India, December 19-21, 2014.
34. The 20th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2014), Berlin, Germany, April 2014.
35. International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2014), Lisbon, Portugal, 7-9 January, 2014.
36. 14th ACM/IFIP/USENIX International Middleware Conference (Middleware 2013), December 9-13, Beijing, China.
37. 32nd International Symposium on Reliable Distributed Systems (SRDS 2013), September 30-October 3, 2013 at Braga, Portugal.

38. 17th International Software Product Line Conference SPLC, Tokyo, Japan, 26-30 August 2013.
39. First International Workshop on Engineering Mobile-Enabled Systems, in conjunction with ICSE 2013, May 18-26th, 2013, San Francisco, CA.
40. International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), Barcelona, Spain, 19-21 February, 2013.
41. ACM/USENIX/IFIP International Middleware conference, Montreal, Quebec, Canada, December 3-7, 2012.
42. 11th Workshop on Adaptive and Reflective Middleware, in conjunction with Middleware 2012 in Montreal, Quebec, Canada, December 3-7, 2012.
43. International Workshop on Real-Time and Distributed Computing in Emerging Applications (REACTION) 2012, San Juan, Puerto Rico, December 4, 2012, in co-location with the 33rd IEEE Real-Time Systems Symposium.
44. Third International Conference on Emerging Applications of Information Technology (EAIT) November 29 - December 01, 2012, Kolkata, India.
45. IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS), Las Vegas, USA, November 12 - 14, 2012.
46. 31st International Symposium on Reliable Distributed Systems (SRDS), 8th-11th October 2012. Irvine, California.
47. Sixth International Workshop on Dynamic Software Product Lines (DSPL), September 2 - 7, 2012, Salvador, Brazil.
48. 16th International Software Product Line Conference (SPLC 2012), Salvador, Brazil on 02-07 September 2012.
49. 5th International workshop UML and Formal Methods (UML&FM 2012), Paris, France, August 27-31, 2012.
50. UML&AADL 2012, July 18-20,2012, Ecole Normale Superieure, Paris, France.
51. 17th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2012), July 18-20,2012, Ecole Normale Superieure, Paris, France.
52. COMPSAC 2012 - Trustworthy Software Systems for the Digital Society, July 16-20, 2012, Izmir, Turkey.
53. Foundations Track of the 8th European Conference on Modelling Foundations and Applications (ECMFA 2012), Copenhagen, Denmark, 2-6th of July, 2012.
54. 24th International Conference on Software Engineering and Knowledge Engineering, Redwood City, California, USA, July 1-3, 2012.
55. 12th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS'12), Stockholm, Sweden, 13-16 June 2012.
56. 15th IEEE International Symposium on Object and component-oriented Real-time distributed Computing (ISORC), April 11-13, 2012, Shenzhen, China.
57. 23rd IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2011), Dallas, USA, December 14 to 16, 2011.
58. Fourth IEEE International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2011), December 12th–14th 2011, University of California, Irvine, CA.
59. ACM/IFIP/USENIX International Middleware Conference, Lisbon, Portugal, December 12th to 16th, 2011.
60. 9th International Conference on Advances in Mobile Computing and Multimedia (MoMM2011), Hue City, Vietnam, 05-07 December 2011.
61. Control Systems, Automation and Robotics track of the 3rd International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT 2011), Hungary on October 5-7, 2011.
62. 15th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2011), August 29th - September 2nd, 2011, Helsinki, Finland.

63. 15th International Software Product Line Conference (SPLC 2011), Research/Experience Track, Munich, Germany, August, 22-26, 2011.
64. 15th International Software Product Line Conference (SPLC 2011), Industry Track, Munich, Germany, August, 22-26, 2011.
65. 2nd Workshop on Formal Methods in Software Product Line Engineering - Munich (Germany), August 2011.
66. 23rd International Conference on Software Engineering and Knowledge Engineering (SEKE2011), Miami Beach, USA, July 7-9, 2011.
67. 2nd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems, July, 5th 2011, Porto, Portugal.
68. Fourth IEEE International workshop UML and Formal Methods, co-located with FM 2011, June 20th, 2011, Lero, Limerick, Ireland.
69. The Software Engineering and Data Engineering (SEDE 2011) conference, Las Vegas, Nevada, June 20-22, 2011.
70. 3rd International Workshop on Model-Driven Architecture and Modeling-Driven Software Development (MDA&MDSD 2011) in conjunction with the 6th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE 2011, Beijing Jiaotong University, 8-11, June 2011.
71. 11th International IFIP Conference on Distributed Applications and Interoperable Systems (DAIS 2011), Reykjavik, Iceland, June 6-9 2011.
72. Second Product LinE Approaches in Software Engineering (PLEASE) workshop, collocated with 33rd International Conference on Software Engineering, Waikiki, Honolulu, Hawaii, May 21-28, 2011.
73. 16th Annual IEEE International Conference on the Engineering of Complex Computer Systems (ICECCS), April 27th-29th, 2011 Las Vegas, NV, USA.
74. Sixth IEEE International workshop UML and AADL, in conjunction with ICECCS 2011, April 27th, 2011, Las Vegas, USA.
75. First International Workshop on Cyber-Physical Networking Systems (CPNS'2011), in conjunction with INFOCOM 2011, April 15, 2011, Shanghai, China.
76. 2nd Workshop on Model Based Engineering for Embedded System Design (M-BED 2011), colocated with the Design, Automation, and Test in Europe (DATE) conference, 14-18, March, 2011, Grenoble, France.
77. Second International Conference on Emerging Applications of Information Technology (EAIT 2011), February, 2011 at Kolkata, India.
78. Fifth International Workshop on "Variability Modeling of Software-intensive Systems" (VaMoS '11), January 27-29 2011 in Namur, Belgium.
79. 9th Workshop on Adaptive and Reflective Middleware (ARM 2010) November 27, 2010, Bangalore India, collocated with Middleware 2010.
80. The 22nd IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2010), November 8-10, 2010, Marina Del Ray, California.
81. International Conference on Software Engineering, Management, and Application (ICSEMA 2010) Kathmandu, Nepal, October 29th and 30th, 2010.
82. The MobiCPS 2010 workshop, held in conjunction with the 7th International Conference on Ubiquitous Intelligence and Computing (UIC2010), October 26-29, 2010 Xian, China.
83. Fourteenth IEEE International Enterprise Computing Conference (EDOC 2010), 25-29 October 2010, Vitoria, ES, Brazil.
84. Advances in Business ICT (ABICT) 2010 Workshop Wisla, Poland, October 18-20, 2010.
85. 3rd Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB), held in conjunction with MoDELS 2010, Oslo, Norway, October 3-8, 2010.
86. 4th Dynamic Software Product Line Workshop held in conjunction with the 14th International Software Product Line Conference 2010, Jeju Island, South Korea, September 13-17, 2010.

87. TOOLS Europe 2010, Malaga, Spain, June 28 to July 2, 2010.
88. 22nd International Conference on Software Engineering and Knowledge Engineering (SEKE'2010), to be held July 1-3, 2010, Redwood City, California.
89. 13th International Symposium on Component Based Software Engineering (CBSE 2010), June 23-25 2010 in Prague, Czech Republic.
90. Sixth European Conference on Modelling Foundations and Applications (ECMFA), University of Pierre & Marie Curie, Paris, France. June 15-18, 2010.
91. 10th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS), Amsterdam, The Netherlands, June 7-9, 2010.
92. The 11th OMG Real-time/Embedded CORBA workshop, Washington DC, May 24-26, 2010.
93. Industrial track at the 32nd International Conference on Software Engineering (ICSE 2010), Cape Town (South Africa), May 2-8, 2010.
94. Thirteeth International Conference on Business Information Systems (BIS 2010), Berlin, Germany, May 3-5 2010.
95. 1st International Workshop on Product LinE Approaches in Software Engineering, May 2nd, 2010, Cape Town, South Africa, held in conjunction with the 32nd International Conference on Software Engineering (ICSE 2010).
96. Workshop on Effective Multicasting for Future Critical Networked Systems (EMFINES 2010), at the Eighth European Dependable Computing Conference (EDCC), Valencia, Spain, April 28-30, 2010.
97. 1st Workshop on Model-Based Engineering for Embedded Systems Design, co-located with DATE 2010, March 12, 2010 in Dresden, Germany.
98. IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2010), Oxford 22-26, March 2010.
99. Special session on "Advanced Peer-to-Peer Protocols and Applications" at the Ninth IASTED International Conference on Parallel and Distributed Computing and Networks (PDCN 2010) February 16-18, 2010 Innsbruck, Austria.
100. Fourth Variability Modelling of Software-intensive Systems (VaMoS '10) workshop, Linz, Austria - January 27-29, 2010.
101. 8th Workshop on Adaptive and Reflective Middleware (ARM'09), in collocation with the 10th ACM/IFIP/USENIX Middleware Conference, in Urbana Champaign, Illinois, November 30th, 2009.
102. Workshop committee for OOPSLA 2009, Orlando Floria, October 25-29, 2009.
103. The ARTIST 2nd International Workshop on Model Based Architecting and Construction of Embedded Systems (ACESMB 2009), in conjunction with the 12th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS 2009), October 6th, 2009, Denver, Colorado.
104. The Thirteenth IEEE International EDOC Conference (EDOC 2009), 31 August - 4 September 2009, Auckland, New Zealand.
105. The 10th OMG Real-time/Embedded CORBA workshop, Washington DC, July 13–15, 2009.
106. The Software Engineering and Knowledge Engineering (SEKE'2009) conference, July 1-3, 2009, Boston, MA.
107. 12th International Symposium on Component-Based Software Engineering (CBSE 2009), East Stroudsburg University, Pennsylvania, USA, June 22-25, 2009.
108. The Second International Workshop on Cyber-Physical Systems (WCPS2009), held in conjunction with IEEE ICDCS 2009 in Montreal, Canada, June 22, 2009.
109. The Fifth European Conference on Model Driven Architecture Foundations and Applications (ECMDA), Gdansk, Poland, summer of 2009.
110. The 9th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2009) conference, Lisbon, Portugal, June 9-11, 2009.

111. The Fourth International Conference on COMmunication System softWAre and middlewaRE (COMSWARE), 15th - 19th June 2009, Trinity College Dublin, Ireland.
112. The UML&AADL Workshop, held in conjunction with ICECCS 2009 The fourteenth IEEE International Conference on Engineering of Complex Computer Systems June 02, 2009, Potsdam, Germany.
113. The 15th Real-time and Embedded Applications Symposium (RTAS) 2009, Track B, Real-time and Embedded Applications, Benchmarks and Tools, San Francisco, CA, United States, April 13 - 16, 2009.
114. Member of the ISORC 2009 advisory and publicity committee for ISORC 2009, March 17-20, 2009, Toyko, Japan.
115. the 13th International Software Product Line Conference (SPLC), August 24-28, 2009, San Francisco, CA.
116. the European Conference on Model Driven Architecture - Foundations and Applications 2009, University of Twente, Netherlands, June 2009.
117. The third workshop on "Variability Modelling of Software-intensive systems" (VaMoS'09), January 28-30 2009 in Sevilla, Spain.
118. the 1st Workshop on Software Reuse Efforts, November 27-29, 2008 Brazil.
119. the 7th Workshop on Adaptive and Reflective Middleware (ARM'08) in collocation with the 9th ACM/IFIP/USENIX Middleware Conference, Leuven, Belgium, December 1st 2008.
120. the Middleware 2008 9th International Middleware Conference, December 1-6, 2008, Leuven, Belgium.
121. the 11th Component-Based Software Engineering conference, Karlsruhe, Germany, October 14-17, 2008.
122. the ARTIST International Workshop on Model Based Architecting and Construction of Embedded Systems (ACESMB 2008), in conjunction with the 11th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS 2008), Toulouse, September 29th, 2008.
123. the 6th Java Technology for Real-Time and Embedded Systems (JTRES) conference, Santa Clara, California, USA, 24-26 September, 2008.
124. the 12th IEEE International Enterprise Distributed Computing Conference (EDOC) (EDOC 2008), 15-19 September 2008, Munich, Germany.
125. the First Workshop on Analyses of Software Product Lines (ASPL'08), September 12, 2008 in Limerick, Ireland in conjunction with SPLC'08.
126. the 9th OMG Real-time/Embedded CORBA workshop, Washington DC, July 14–17, 2008
127. the 3rd International Conference on Software and Data Technologies, July 5-8, 2008, Porto, Portugal.
128. the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE'08), Redwood City, California, USA, July 1-3, 2008.
129. the TOOLS EUROPE 2008 conference, June 30 to July 4, 2008 at ETH Zurich.
130. National Conference on Research & Development in Hardware & Systems (CSI-RDHS 2008), Computer Society of India Kolkata Chapter & CSI Division I (Hardware & Systems), June 20-21, 2008, Kolkata, India.
131. the First International Workshop on Cyber-Physical Systems, Beijing, China, June 17 - 20, 2008.
132. the ECMDA 2008 (Fourth European Conference on Model Driven Architecture Foundations and Applications) in Berlin, June 09 - 12, 2008.
133. the Distributed Applications and Interoperable Systems (DAIS), Oslo, Norway, June 4, 2008.
134. the 2nd International Workshop on Ultra-Large-Scale Software-Intensive Systems (ULSSIS 2008), May 10-11, 2008 Leipzig, Germany.
135. the Automotive Systems Track at the 30th International Conference on Software Engineering (ICSE), Leipzig, Germany, 10-18 May 2008.

136. the Real-Time and embedded Applications / Benchmarks track at the 14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2008), St. Louis, MO, April 22-24, 2008.
137. the 3rd UML and AADL Workshop held in conjunction with the 13th IEEE International Conference on Engineering of Complex Computer Systems, Belfast, Northern Ireland, 31 March - 4 April 2008.
138. the ACM Programming for Separation of Concern track at SAC 2008, Fortaleza, Brazil, March 16 - 20, 2008.
139. the 6th edition of the International Workshop on Adaptive and Reflective Middleware, held in conjunction with Middleware 2007 in Newport Beach, California.
140. the IEEE/ACM/USENIX Middleware conference, November 2007.
141. the IASTED International Conference on Parallel and Distributed Computing and Systems, PDCS 2007, Cambridge, MA, USA from Nov 19-21, 2007.
142. the 9th International Symposium on Distributed Objects, Middleware, and Applications (DOA), Iberian peninsula and islands, Oct 28 - Nov 2, 2007.
143. Member of the Doctoral Symposium committee at OOPSLA 2007, Portland, OR October 21-25, 2007.
144. the International Symposium on Ambient Intelligence and Computing, October 2007, Korea.
145. the IEEE conference on Enterprise Distributed Object Computing (EDOC), Annapolis, MD, October 15-19, 2007.
146. the 5th Java Technology for Real-Time and Embedded Systems (JTRES), Vienna, Austria, 26-28 September, 2007.
147. the Workshop on Trade-Off analysis of Software Quality Attributes (TOSQA), collocated with the sixth joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, Dubrovnik, Croatia, September 3-7, 2007.
148. the 2nd International Conference on Software and Data Technologies, July 22-25th 2007, Barcelona, Spain.
149. the Fourth IEEE International Conference on Web Services, Salt Lake City, UT, July 9-13, 2007.
150. the 10th International Component-Based Software Engineering (CBSE) Symposium, Boston, MA, July 9-11 2007.
151. the 8th OMG Real-time/Embedded CORBA workshop, Washington DC, July 9–12, 2007.
152. the International Conference TOOLS EUROPE 2007, Zurich, Switzerland on June, 24-28 2007.
153. the track on "Real-Time and Embedded Applications and Benchmarks" for the 13th IEEE Real-Time and Embedded Technology and Applications Symposium, Bellevue, WA, April 3 - April 6, 2007.
154. the Workshop on the Foundations of Interactive Computation (FInCo 2007), Braga, Portugal, March 24 - April 1, 2007.
155. the 15th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), Long Beach, California, 26-27 March, 2007.
156. the ACM Symposium on Applied Computing, Programming for Separation of Concerns track, Seoul, Korea, March 11 - 15, 2007.
157. the Workshop on Pervasive Computing Environments and Services (PCES 07), Naples, Italy, Feb 7-9, 2007.
158. the Minitrack on Components for Embedded and Real-time Systems at the 40th Hawaiian International Conference on System Sciences, January 3-6, 2007 at Waikoloa, Big Island, Hawaii.
159. the 13th Asia Pacific Software Engineering Conference (APSEC06), Bangalore, India, Dec 6-8, 2006.
160. the Real-time Middleware and Software Engineering track of the The 27th IEEE Real-Time Systems Symposium, December 5-8, 2006 Rio de Janeiro, Brazil.

161. the 2nd International Conference on Trends in Enterprise Application Architecture, November 29th to December 1st, 2006, Berlin, Germany.
162. the workshop on MOdel Driven Development for Middleware (MODDM), November 27, 2006, Melbourne, Australia.
163. the International Symposium on Distributed Objects and Applications (DOA), Montpellier, France, Oct 29 - Nov 3, 2006.
164. the "Library-Centric Software Design" (LCSD'06) workshop at the OOPSLA'06 conference in Portland, Oregon, October 22-26, 2006.
165. Judge for the Student Research Competition at OOPSLA 2006, Portland, OR, October 23-24, 2006.
166. the NSF Workshop On Cyber-Physical Systems, October 16 - 17, 2006, Austin, Texas.
167. the Models at Run-Time MaRT-06 workshop held at the MoDELS 2006 conference, Genova Italy, Oct. 2-6, 2006.
168. the MoDELS 2006 conference, Genova Italy, Oct. 2-6, 2006.
169. the 7th OMG Real-time/Embedded CORBA workshop, Washington DC, July 11–14, 2006.
170. the European Conference on Object-Oriented Programming, Nantes, France, July 3-7, 2006.
171. the 9th International Symposium on Component-Based Software Engineering (CBSE 2006), Malardalen University, Sweden, June 29th-1st July 2006.
172. the 28th International Conference on Software Engineering (ICSE 28), May 24-26, 2006, Shanghai, China.
173. the 14th International Workshop on Parallel and Distributed Real-Time Systems, April 25-26, 2006, Island of Rhodes, Greece.
174. the 9th IEEE International Symposium on Object-oriented Real-time Distributed Computing, April 24-26, 2006, Gyeongju, Korea.
175. the Automotive Software Workshop San Diego (ASWSD 2006), University of California, San Diego, March 15-17, 2006.
176. the C++ Connections: 20 Years of C++ conference, Nov 7-11, 2005, Mandalay Bay, Las Vegas, NV.
177. the Conference on Distributed Objects and Applications (DOA 2005), Oct 31 - Nov 4, 2005, Agia Napa, Cyprus.
178. the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOSPLA), October 16-20, 2005, San Diego, CA, USA.
179. the 6th International Conference on Middleware (Middleware'2005), October, 2005, Grenoble, France.
180. the 2005 Monterey Workshop on Networked Systems, Laguna Beach, California, September 22-24, 2005.
181. The 12th Pattern Language of Programs (PLoP 2005), September 7-10, 2005, Allerton Park, Monticello, Illinois, USA.
182. the 14th IEEE International Symposium on High-Performance Distributed Computing (HPDC-14), Research Triangle Park, North Carolina, July 27, 2005.
183. the 5th International Workshop on Software and Performance (WOSP 2005), Palma de Mallorca, Spain, July 11-15, 2005.
184. the 6th OMG Real-time/Embedded CORBA workshop, Washington DC, July 11–14, 2005.
185. the 5th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS 2005), June 15-17, 2005, Athens, Greece.
186. the International Conference on Autonomic Computing (ICAC 2005), Seattle, WA, June 2005.
187. the International Symposium on Component-Based Software Engineering (CBSE), co-located with the International Conference on Software Engineering (ICSE), May 14-15, 2005, St. Louis, MO.

188. the Foundations of Interactive Computation (FINCO'05) Workshop, Saturday, 9 April 2005, in Edinburgh, Scotland.
189. the Embedded Applications track of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) 2005, San Francisco, California, March 2005.
190. the "Programming for Separation of Concerns" track at Symposium on Applied Computing (SAC 2005), Santa Fe, New Mexico, March 2005.
191. the 12th International Symposium on the Foundations of Software Engineering, November 6th, 2004, Newport Beach, California.
192. the Conference on Distributed Objects and Applications (DOA 2004), October 25-29, 2004 in Cyprus, Greece.
193. the 2nd International Workshop on Java Technologies for Real-Time and Embedded Systems (JTRES), October 25-29, 2004, Larnaca, Cyprus.
194. the 3rd Workshop on Reflective and Adaptive Middleware (RM2004), October 19, 2004, Toronto, Ontario, Canada.
195. the Middleware 2004 5th IFIP/ACM/USENIX International Conference on Distributed Systems Platforms, October 18-22, 2004, Toronto, Canada.
196. the 4th TAO+CIAO Workshop, Arlington, VA, July 16, 2004.
197. the DARPA Workshop on Java in Real-Time and Embedded Defense Applications, Arlington, VA, July 13, 2004.
198. the OMG Real-time/Embedded CORBA workshop, Crystal City, VA, July 12–15, 2004.
199. the ECOOP 2004 conference, June 14-18, 2004, Oslo, Norway.
200. the Middleware track of the 24th IEEE International Conference on Distributed Computing Systems (ICDCS), May 23-26, 2004, Tokyo, Japan.
201. the 2nd International Workshop on Remote Analysis and Measurement of Software Systems (RAMSS), Edinburgh, Scotland, UK, May 24, 2004.
202. Aspect-Oriented Software Development conference, Lancaster, England, March 22-26, 2004.
203. the SPIE/ACM Conference on Multimedia Computing and Networking, January 21-22, 2004 Santa Clara, California.
204. the Real-time Systems Symposium (RTSS), Cancun, Mexico, December 3-5, 2003.
205. the 4th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS), Paris - France November 17-21, 2003.
206. the International Workshop on Java Technologies for Real-Time and Embedded Systems (JTRES), November 3-7, 2003, Catania, Sicily, Italy.
207. the Domain Driven Development track at the OOPSLA 2003 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, October 26-30, 2003, Anaheim, California, USA.
208. the OOPSLA 2003 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, October 26-30, 2003, Anaheim, California, USA.
209. External reviewer for the 2nd Generative Programming and Component Engineering (GPCE '03) conference, Erfurt, Germany, September 22-25, 2003.
210. the OMG Real-time/Embedded CORBA workshop, Crystal City, VA, July 14–17, 2003.
211. the The 2nd Workshop on Reflective and Adaptive Middleware, Rio de Janeiro, Brazil, June 17, 2003.
212. the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation (PLDI), San Diego, California, June 9 - 11, 2003.
213. the 1st International Workshop on Remote Analysis and Measurement of Software Systems (RAMSS), Portland, Oregon, May 9, 2003.
214. External reviewer for the 17th International Parallel and Distributed Processing Symposium, April 22–26, 2003, Nice, France.

215. the ACM International Conference on Aspect-Oriented Software Development, March 17 - 21, 2003, Boston, MA.
216. the SPIE/ACM Conference on Multimedia Computing and Networking, Santa Clara, California, January 29–31, 2003.
217. the International Workshop on Product Line Engineering The Early Steps: Planning, Modeling, and Managing (PLEES '02), Seattle, WA, November 5, 2002.
218. the 8th IEEE Real-Time and Embedded Technology and Application Symposium (RTAS), San Jose, CA, September 24-27, 2002.
219. the 9th Conference on Pattern Language of Programs, Allerton Park, IL, September 8-12, 2002.
220. the Workshop on Dependable Middleware-Based Systems, held as a part of DSN 2002, Washington, D.C., June 23-36, 2002.
221. the 2nd TAO Workshop, Arlington, VA, July 19, 2002.
222. the OMG Real-time/Embedded CORBA workshop, Crystal City, VA, July 15–18, 2002.
223. the 16th European Conference on Object-Oriented Programming, University of Malaga, Spain June 10-14, 2002.
224. the Tenth International Workshop on Quality of Service (IWQoS), May 15-17, 2002, Miami Beach, Florida.
225. the International Symposium on Object-Oriented Real-time Distributed Computing (ISORC), Washington DC, April 29 – May 1, 2002.
226. the Seventh IEEE International Workshop on Object-oriented Real-time Dependable Systems (WORDS 2002), January 7-9, 2002, San Diego, CA.
227. the International Workshop on Multimedia Middleware October 5th, 2001, Ottawa, Canada.
228. the OMG Workshop on Real-time and Embedded CORBA, in Reston, VA, June 4-6, 2001.
229. the USENIX 2001 conference, Boston, MA, June 25-30, 2001.
230. the International Symposium on Object-oriented Real-time Distributed Computing (ISORC), May 2-4, Magdenburg, Germany, 2001.
231. the 6th USENIX Conference on Object-Oriented Technologies and Systems, January 27 - February 3, 2001, San Antonio, TX.
232. External reviewer for OOPSLA 2000, Minneapolis, MN, October 2000.
233. the 3rd IFIP International Conference on Trends towards a Universal Service Market (USM'2000), September 12-14, 2000.
234. the International Symposium on Distributed Objects and Applications (DOA '00), OMG, Antwerp, Belgium, September 2000.
235. the ACM SIGCOMM 2000, Stockholm, Sweden, August 30 to September 1st, 2000.
236. the Pattern Languages of Programming (PLoP) conference, Monticello, Illinois, August, 2000.
237. the 9th IEEE International Conference on High-Performance Distributed Computing, August, 2000.
238. the "International Workshop on Software Engineering for Parallel and Distributed Systems" (PDSE 2000), at the 22nd International Conference on Software Engineering (ICSE-2000), in Limerick, Ireland in June, 2000.
239. the 6th IEEE Real-Time Technology and Application Symposium (RTAS), May 17-19, 2000, Washington DC, USA.
240. the 1999 ACM OOPSLA conference, Denver, Colorado, November 1-5, 1999.
241. the IFIP Sixth International Workshop on Protocols For High-Speed Networks (PfHSN '99), Wednesday August 25 – Friday August 27, 1999 Salem, MA.
242. the 1999 IEEE Real-Time Technology and Applications Symposium (RTAS99), Vancouver, British Columbia, Canada, June 2-4, 1999.
243. the 5th USENIX Conference on Object-Oriented Technologies and Systems, May 3-7, 1999, San Diego, CA.

244. Technical workshop committee for the International Software Architecture workshop, ACM SIGSOFT's FSE9 conference in Orlando FL, November 1-5, 1998.
245. the workshop on Software and Performance (WOSP98), Santa Fe, New Mexico, Oct 12-16 1998.
246. the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing: Middleware '98. September 15-18 1998, The Lake District, England.
247. the TOOLS USA'98 conference. Santa Barbara, California, August 3 - 7, 1998.
248. the IEEE High Performance Distributed Computing conference, Chicago, IL, July 28-31, 1998.
249. 12^{th} European Conference on Object-Oriented Programming, Brussels, Belgium, July 20 - 24, 1998.
250. the 3rd EuroPLoP conference, Kloster Irsee, Germany, July 9-11, 1998.
251. the IEEE International Conference on Configurable Distributed Systems (ICCDS '98), Annapolis, MD, May 4-6, 1998.
252. the IEEE IWQoS '98 in Napa Valley, CA, May 18-20, 1998.
253. the 4th USENIX Conference on Object-Oriented Technologies and Systems, April 26-29, 1998, Santa Fe, New Mexico.
254. the 3^{rd} International Workshop on Software Engineering for Parallel and Distributed Systems, at the 20th International Conference on Software Engineering (ICSE-20), in April 20-21, Kyoto, Japan.
255. the IEEE Conference on Open Architectures and Network Programming, April 3-4, 1998, San Francisco, CA.
256. the Workshop on Middleware for Real-Time Systems and Services, held in conjunction with IEEE Real-time Systems Symposium, December 2nd, San Francisco, California.
257. the Open Signaling for ATM, Internet and Mobile Networks. October 6th and 7th, 1997, Columbia University, New York, NY.
258. the 24^{th} International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Asia '97). Beijing, China, September 22 - 25, 1997.
259. the 4^{th} Pattern Languages of Programming conference, Allerton Park, Illinois, September 3-5, 1997.
260. the 3^{rd} USENIX Conference on Object-Oriented Technologies and Systems, Portland, June 16-19th 1997.
261. Session chair of the Patterns technical paper session at ECOOP '97, June 13th, 1997.
262. the 1997 European Conference on Object-Oriented Programming (ECOOP), June 9-13, 1997, Jyvskyl, Finland.
263. Chair of the technical session on "Distributed Object Computing" for the IFIP/IEEE Fifth International Workshop on Quality of Service (IWQoS '97).
264. the 2^{nd} International Workshop on Software Engineering for Parallel and Distributed Systems, at the 19^{th} International Conference on Software Engineering (ICSE-19) Sheraton Boston Hotel and Towers, Boston, Massachusetts, USA, May 19 and 20, 1997.
265. the 3^{rd} USENIX Conference on Object-Oriented Technologies and Systems, Portland, 1997.
266. the 5^{th} IEEE International Workshop on Object-Orientation in Operating Systems, IEEE TCOS and USENIX, Seattle, Washington, October 27-28, 1996.
267. the 1997 ACM SIGCOMM conference, Cannes, French Riviera, France, September 1997.
268. the 1997 IEEE INFOCOM conference, Kobe, Japan, April 1997.
269. the 1996 IEEE INFOCOM conference, San Francisco, CA, USA, March 24-28, 1996.
270. the 1995 IEEE INFOCOM conference, Boston, Massachusetts, USA, April, 1995.
271. the 3^{rd} IEEE workshop on Architecture and Implementation of High Speed Communication Subsystems (HPCS '95), held in Mystic, Connecticut, August 1995.
272. the 8^{th} IFIP International Working Conference on Upper Layer Protocols, Architectures, and Applications, held in Barcelona, Spain, June 1 to 3, 1994.

Workshops and Panels Organized

1. Co-organized the 1st International Workshop on Data Dissemination for Large scale Complex Critical Infrastructures (DD4LCCI 2010), at the Eighth European Dependable Computing Conference, Valencia, Spain, April 28-30, 2010.
2. Co-organized the OOPSLA Jeopardy panel at OOPSLA 2009, Orlando Floria, October 25-29, 2009.
3. Co-organized a workshop entitled First International Workshop on Software Technologies for Ultra-Large-Scale (ULS) Systems at 29th Int. Conference on Software Engineering, May 20-29th, Minneapolis, MN, 2007.
4. Co-organized a session on architectures, platforms, and standards for QoS-enabled dissemination at the Systems and Information Interoperability Meeting, Oct 25-27, 2006 at the Minnowbrook Conference Center, Blue Mountain Lake, NY.
5. Co-organized a workshop entitled "Breathturn: Ultra Large Scale Systems" at OOPSLA 2006, October 26, 2006, Portland, OR.
6. Co-chair of the NSF workshop on open-source Middleware for Distributed Real-time and Embedded Systems, 7th OMG Real-time/Embedded CORBA workshop, Arlington, VA, July 10–13, 2006.
7. Organized and led a session on architectures, platforms, and standards for real-time tactical information management at the Systems and Information Interoperability Meeting, Oct 18-21, 2005 at the Minnowbrook Conference Center, Blue Mountain Lake, NY.
8. Co-organizer of the technical workshops program at OOPSLA 2005, San Diego, October 16th-20, 2005.
9. Co-organizer for the MODELS 2005 workshop on "MDD for Software Product-lines: Fact or Fiction?," October 2, 2005, Jamaica.
10. Co-organizer of the OOPSLA '02 workshop on "Patterns in Distributed Real-Time and Embedded Systems", Seattle, WA, November, 2002.
11. Co-organizer of the OOPSLA '01 workshop on "Towards Patterns and Pattern Languages for OO Distributed Real-time and Embedded Systems" Tampa Bay, FL, October 14, 2001.
12. Organizer and chair of a panel on real-time extensions to OO middleware, OPENSIG Fall '97 workshop on Open Signaling for ATM, Internet and Mobile Networks Columbia University, October 6-7 1997, New York, NY.
13. Co-organizer of a workshop for the 1997 European Conference on Object-Oriented Programming entitled CORBA: Implementation, Use, and Evaluation, Jyvaskyla, Finland, June 10th, 1997.
14. Organizer and chair of a panel on "QoS and Distributed Systems Platforms" for the IFIP Fifth International Workshop on Quality of Service (IWQoS '97), May 22-24th, 1997, Columbia University, New York.
15. Co-organizer of the OOPSLA '95 workshop on "Patterns for Concurrent, Parallel, and Distributed OO Systems."
16. Co-facilitator of the ECOOP '95 workshop workshop on Pattern Languages of Object-Oriented Programs, Aarhus, Denmark, August 1995.

Reviewer for Professional Submittals

Reviewed papers for the following journals, conferences, books, and grant review processes:

1. Reviewer for COVID-19 proposals to the C3.ai Digital Transformation Institute.
2. *The 21st IEEE International Symposium on Real-time Computing (ISORC)*, Nanyang Technological University, Singapore, 29th - 31st May 2018.
3. *Future Generation Computer Systems*, Elsevier, edited by Aniruddha Gokhale et al., 2016.
4. *IEEE Software*, Special Issue on Next Generation Mobile Computing, edited by James Edmondson et al., 2013.
5. *Software Testing in the Cloud*, edited by Scott Tilley, 2012.
6. Elsevier Information & Software Technology special issue on Software Reuse and Product Lines, 2012.
7. The 2010 Military Communications Conference, Cyber Security and Network Management, San Jose, CA, October 31-November 3, 2010.

8. *Model-Driven Domain Analysis and Software Development: Architectures and Functions*, edited by Janis Osis and Erika Asnina, 2010.
9. Reviewer for the book "Patterns for Parallel Software Design," by Jorge L. Ortega Arjona, Wiley, 2010.
10. Special Issue on Industrial Applications of Aspect Technology for the journal Transactions on Aspect-Oriented Software Development (TAOSD), 2009.
11. *Software Engineering for Self-Adaptive Systems*, edited by Betty H. C. Cheng, Rogerio de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, Springer, 2009.
12. Special issue on Service Oriented Computing for the ACM Transactions on the Web journal, 2008.
13. Special Issue in Software Reuse: Methods, Processes, Tools and Experiences for the Journal of the Brazilian Computer Society (JBCS), 2007
14. Designing Software-Intensive Systems: Methods and Principles book, 2008
15. Special issue on Patterns for the IEEE Software, 2007
16. IEEE Internet Computing Magazine, 2006.
17. IEEE Transactions on Parallel and Distributed Systems, 2004
18. International Journal of Software Process: Improvement and Practice Special issue - Software Variability: Process and Management
19. IEEE Internet Computing Magazine
20. 2004 NSF NSG panel
21. IEEE Transactions on Parallel and Distributed Computing special issue on Middleware, 2003
22. 2003 NSF ITR panel
23. 2002 NSF CAREER panel
24. IEEE Internet Computing Magazine, 2002
25. NIST Competence Proposals, May 2002
26. DARPA MoBIES program, May 2002
27. DARPA NEST program, May 2002
28. DARPA DASADA program, April 2002
29. Elsevier Journal of Systems and Software Special Issue on Software Architecture: Engineering Quality Attributes, 2002
30. IEEE Communications Magazine, Evolving Communications Software: Techniques and Technologies, 2001
31. DARPA Network Embedded Software Technology (NEST) program, 2001
32. DARPA Software Enabled Control (SEC) program, 2000
33. IEEE Concurrency magazine, Object-Oriented Systems Track, 1999
34. IEEE Journal on Selected Areas in Communications special issue on "Service Enabling Platforms for Networked Multimedia Systems," 1999
35. IEEE Journal of Communications and Networks, 1999
36. Reviewer for the 4^{th} Pattern Languages of Programming Design book published by Addison Wesley
37. The International Journal of Time-Critical Computing Systems, special issue on Real-time Middleware, edited by Wei Zhao
38. Next Generation Internet (NGR) networking research review panel, October 1998
39. IEE Transactions on Software Engineering, special issue on Configurable Distributed Systems
40. Theme issue on Symbolic Modeling in Practice for the Communications of the ACM
41. "Multimedia DBMS and the WWW" Minitrack at the 32nd Hawaii International Conference on System Sciences, 1999
42. "Dependable Distributed Systems" Minitrack at the 32nd Hawaii International Conference on System Sciences, 1999

43. IEEE Computer special issue on "Design Challenges for High-Performance Network Interfaces," 1998
44. 1998 NSF Experimental Software Systems review panel.
45. ACM SIGMetrics Conference, 1998
46. ACM Transactions on Software Engineering Methods
47. Special Issue on Patterns and Pattern Languages for the journal of Theory and Practice of Object Systems, (Stephen P. Berczuk, Editor), John Wiley and Sons, 1995
48. Special Issue of Computer Communications on Building Quality of Service into Distributed Systems
49. IEEE Communications Magazine
50. IEEE/ACM Journal of Transactions on Networking
51. Communications of the ACM
52. IEE/BCS Distributed Systems Engineering Journal
53. Software Practice and Experience, John Wiley and Sons
54. 1998, 1997, and 1996 NSF networking program
55. 1996 NSF software engineering and programming languages CAREER panel
56. 1994 California MICRO (Microelectronics Innovation Computer Research Opportunity) engineering computer network grant review process
57. IEEE Conference on Parallel and Distributed Computing Systems, 1994
58. IEEE International Conference on Computer Communications and Networks, 1994
59. IEEE INFOCOM conference, 1994
60. 1993 NASA Applied Information Systems Research grant review process
61. 1992 California MICRO (Microelectronics Innovation Computer Research Opportunity) engineering computer network grant review process
62. 7^{th} *IFIP International Conference on Upper Layer Protocols, Architectures, and Applications*, 1992
63. The 1992 Special Issue on Measurement for IEEE Journal Transactions on Software Engineering

Memberships: IEEE, ACM, and USENIX

Patents

1. US patent 7,523,471 – "Interpretive network daemon implemented by generic main object," in conjunction with Karlheinz Dorn, Dieter Quehl, Detlef Becker, and Christian Scharf of SIEMENS Medical Engineering, Erlangen, Germany, 2009.

Theses Supervised

- *Doctoral and Masters Committees Chaired*
    1. Chaired the masters thesis committee for Cici Wang, November 2021.
    2. Chaired the masters thesis committee for Evan Segaul, March 2021.
    3. Co-chair of the doctoral dissertation defense for Peng Zhang, August 2018.
    4. Co-chair of the doctoral dissertation defense for James Edmondson, March 2012.
    5. Co-chair of the doctoral topic defense for James Edmondson, December 2011.
    6. Co-chair of the doctoral dissertation defense for Will Otte, November 2011.
    7. Chair of the doctoral dissertation defense for Brian Dougherty, March 2011.
    8. Chair of the doctoral topic defense for Brian Dougherty, June 2010.
    9. Chair of the masters defense for Pooja Varshneya, May 2010.
    10. Chair of the doctoral topic defense for Nilabja Roy, March 2010.
    11. Chair of doctoral topic defense for Joe Hoffert, November 2009.

12. Chair of the doctoral dissertation defense for Jai Balasubramanian, September 2009.
13. Chair of masters defense for Friedhelm Wolf, March 2009.
14. Chair of the doctoral dissertation defense for Nishanth Shankaran, October 2008.
15. Chair of the doctoral dissertation defense for Jules White, October 2008.
16. Chair of doctoral dissertation defense for Gan Deng, December 2007.
17. Chair of doctoral dissertation defense for Krishnakumar Balasubramanian, September 2007.
18. Chair of the doctoral topic defense for Nishanth Shankaran, April 2007.
19. Chair of doctoral topic defense for Krishnakumar Balasubramanian, March 2006.
20. Chair of doctoral topic defense for Gan Deng, March 2006.
21. Chair of final doctoral dissertation defense for Arvind Krishna, December 2005.
22. Chair of masters thesis committee for Emre Turkay, summer 2005.
23. Chair of doctoral topic defense for Arvind Krishna, summer 2005.
24. Chair of masters thesis committee for Ossama Othman, December, 2002.
25. Chair of doctoral dissertation committee for Carlos O'Ryan, May, 2002.
26. Chair of dissertation topic defense committee for Carlos O'Ryan, September, 2001.
27. Chair of masters committee for Nagarajan Surendran, August, 1999.
28. Chair of masters committee for Alexander Babu Arulanthu, July, 1999.
29. Chair of oral exam committee for Chris Gill, June, 1999.
30. Chair of doctoral exam committee for Andy Gokhale, May, 1998.
31. Chair of masters exam committee for Sumedh Mungee, May, 1998.
32. Chair of masters exam committee for Sergio Flores, May, 1998.
33. Chair of masters committee for Prashant Jain, June 1997.
34. Chair of doctoral topic defense for James Hu, February 1997.
35. Chair of masters committee for Tim Harrison, February 1997.
36. Chair of doctoral topic defense committee for Andy Gokhale, October, 1996.

- *Doctoral and Masters Committees Member*

1. Served on the doctoral topic defense for Zhongwei Teng, April 2021.
2. Served on the masters thesis committee for Gabriela Gresenz, March 2021.
3. Served on the masters thesis committee for Xiaoxing Qiu, March 2021.
4. Served on the doctoral dissertation defense for Anirban Bhattacharjee, January 2020.
5. Served on the doctoral topic defense for Anirban Bhattacharjee, April 2019.
6. Served on the doctoral dissertation defense for Shunxing Bao, September 2018.
7. Served on the doctoral dissertation defense for Shashank Shekhar, May 2018.
8. Served on the doctoral dissertation defense for Fangzhou Sun, March 2018.
9. Served on the doctoral topic defense for Shunxing Bao, March 2018.
10. Served on the doctoral topic defense for Peng Zhang, January 2018.
11. Served on the doctoral dissertation defense for Marcelino Rodriguez-Cancio, December 2017.
12. Served on the doctoral dissertation defense for Yao Pan, November 2017.
13. Served on the doctoral topic defense for Fangzhou Sun, September 2017.
14. Served on the doctoral topic defense for Shashank Shekhar, May 2017.
15. Served on the doctoral topic defense for Yao Pan, February 2017.
16. Served on the doctoral dissertation defense for Faruk Caglar, July 2015
17. Served on the doctoral dissertation defense for Wei Yan, May 2015.
18. Served on the doctoral dissertation defense for Kyoungho An, March 2015.
19. Served on the masters thesis committee for Songtao Hei, March 2015.
20. Served on the masters thesis committee for Meng Wang, March 2015.
21. Served on the doctoral dissertation defense for Sean Hayes, January 2015.
22. Served on the doctoral dissertation defense for Hamilton Turner, November 2014.
23. Served on the doctoral topic defense for Faruk Caglar, November 2014.

24. Served on the doctoral topic defense for Hamilton Turner, February 2014.
25. Served on the doctoral dissertation defense for Fan Qui, February 2014.
26. Served on the doctoral dissertation defense for Xiaowei Li, May 2013.
27. Served on the doctoral topic defense for Fan Qiu, April 2013.
28. Served on the doctoral dissertation defense for Janos Mathe, August 2012.
29. Served on the doctoral dissertation defense for Tripti Saxena, July 2012.
30. Served on the doctoral dissertation defense for Akshay Dabholkar, April 2012.
31. Served on the doctoral topic defense for Xiawei Li, March 2012.
32. Served on the doctoral topic defense for Janos Mathe, August 2011.
33. Served on the doctoral dissertation defense for Liang Dai, April 2011.
34. Served on the doctoral dissertation defense for Daniel Balasubramanian, March 2011.
35. Served on the doctoral topic defense for Will Otte, February 2011.
36. Served on the doctoral topic defense for Akshay Dabholkar, February 2011.
37. Served on the doctoral dissertation defense for Joe Hoffert, February 2011.
38. Served on the doctoral topic defense for Tripti Saxena, January 2011.
39. Served on the doctoral dissertatin defense for Nilabja Roy, November 2010
40. Served on the doctoral topic defense for Daniel Balasubramanian, October 2010.
41. Served on the doctoral dissertation defense for Sumant Tambe, September 2010.
42. Served on the doctoral topic defense for Sumant Tambe, April 2010.
43. Served on the doctoral dissertation defense for John Kinnebrew, March 2010.
44. Served on the doctoral dissertation defense for Shanshan Jiang, November 2009.
45. Served on the doctoral dissertation defense for James Hill, March 2009.
46. Served on the doctoral topic defense for James Hill, October 2008.
47. Served on the doctoral topic defense for Jai Balasubramanian, August 2008.
48. Served on the doctoral topic defense for Liang Dai, December 2008.
49. Served on the doctoral topic defense for Shanshan Jiang, November 2008.
50. Served on the doctoral topic defense for Jules White, April 2008.
51. Served on the doctoral topic defense for Amogh Kavimandan, February 2008.
52. Served on the doctoral dissertation defense for Amogh Kavimandan, November 2008.
53. Served on the doctoral topic defense for Amogh Kavimandan, February 2008.
54. Served on the doctoral dissertation defense for Michael Stal, University of Groningen, March 2007.
55. Served on the doctoral topic defense for Karlkim Suwanmongkol, fall 2004.
56. Served on the doctoral dissertation topic defense committee for Aditya Agrawal, July, 2004.
57. Served on the doctoral dissertation defense for Angelo Corsaro, July 2004.
58. Served on the doctoral dissertation defense for Nanbor Wang, April 2004.
59. Served on the doctoral topic defense for Angelo Corsaro, October 2003.
60. Served on the doctoral dissertation defense committee for Jonathan Sprinkle, July, 2003.
61. Served on the doctoral dissertation topic defense committee for Aditya Agrawal, June, 2003.
62. Served on masters committee for Kirk Kelsey, March 2003.
63. Served on the dissertation topic defense committee for Jonathan Sprinkle, February, 2003.
64. Served as external examiner for Bob Jolliffe's masters thesis Department of Computer Science, University of South Africa, March, 2003.
65. Served on the doctoral dissertation committee for Irfan Pyarali, December, 2001.
66. Served on the doctoral dissertation committee for Chris Gill, December, 2001.
67. Served as external examiner for Daniel Heggander's Ph.D. dissertation in the Department of Software Engineering and Computer Science at Blekinge Institute of Technology, Sweden, September, 2001.
68. Served as external examiner for Mohammad Radaideh's masters thesis in the Electrical Engineering department at McMaster's University, Canada, Winter 2000.

69. Served as external examiner for David Holmes' Ph.D. dissertation in the information and computer sciences department at Macquarie University, Sydney, Fall 1999.
70. Served on final doctoral dissertation committee for Priya Narasimhan, August, 1999.
71. Served on the doctoral final dissertation defense for Christo Papadopoulos, August, 1999.
72. Served on dissertation topic defense for Michael Plezbert, February, 1999.
73. Served on masters committee for Craig Nauman, February, 1999.
74. Served on the doctoral exam committee for Chuck Cranor, July, 1998.
75. Served on masters exam committee for Mihai Tutunaru, April, 1998.
76. Served on the doctoral exam committee for Michael Plezbert, June, 1997.
77. Served on masters committee for Todd Rogers, June 1997.
78. Served on masters committee for Robert Engel, January 1997.
79. Served on committee for final doctoral dissertation defense of R. Gopalakrishnan, November, 1996.
80. Served on committee for final doctoral dissertation defense of Lorrie Cranor, September, 1996.
81. Served on the doctoral dissertation topic proposal committee for Christos Papadopoulos July, 1995.
82. Served on the doctoral dissertation topic proposal committee for Charles Cranor December, 1994.
83. Served on oral exam committee for Andy Gokhale December, 1994.
84. Served on the doctoral dissertation proposal committee for Lorrie Cranor, December, 1994.
85. Served on the doctoral final dissertation defense committee for Donald Wilcox, November, 1994.
86. Served on masters committee for Madhavapeddi Shreedhar, September, 1994
87. Served on the doctoral dissertation topic proposal committee for R. Gopalakrishnan, September, 1994.

- *Doctoral Student Advisees and Co-Advisees*

1. Mike Walker (USA)

- *Graduated PhD Students*

1. Jaiganesh Balasubramanian, Ph.D., 2009, currently works for Citigroup, New York, NY.
2. Krishnakumar Balasubramanian, Ph.D., 2007, Mathworks, Boston, MA.
3. Angelo Corsaro, Ph.D. 2004, PrismTechnologies, Parise France.
4. Gan Deng, Ph.D., 2007, Citigroup, Charleston, SC.
5. Brian Dougherty, Ph.D. 2011, Optio Labs, Nashville, TN.
6. James Edmondson, Ph.D., 2012, Member of the Technical Staff, Software Engineering Institute, Pittsburgh, PA.
7. Chris Gill, Ph.D. 2001, Professor, Washington University, St. Louis, MO.
8. Andy Gokhale, Ph.D. 1998, Associate Professor, Vanderbilt University, Nashville, TN.
9. James Hill, Ph.D., 2009, Assistant Professor, Indiana University, Purdue University, Indianapolis.
10. Joe Hoffert, Ph.D. 2011, Assistant Professor, University of Edmonton, Canada.
11. John Kinnebrew, Ph.D., 2010, ISIS, Nashville, TN.
12. Arvind Krishna, Ph.D. 2005, Qualcomm, San Diego, CA.
13. Irfan Pyarali, Ph.D. 2001, CitiGroup, New Jersey.
14. Nilabja Roy, Ph.D. 2011, Research Scientist, Institute for Software Integrated Systems, Nashville, TN.
15. Carlos O'Ryan, Ph.D., 2002, CitiGroup, Charleston, SC.
16. Nishanth Shankaran, Ph.D., 2008, Amazon, Seattle, WA.
17. Nanbor Wang, Ph.D. 2004, Research Scientist, Tech-X, Boulder, Colorado.
18. Jules White, Ph.D. 2008, Assistant Professor, Virginia Tech, Blackburg, VA.

- *Graduated Masters and Ugrad Students*

  1. Alexander Babu Arulanthu, MS 1999, Sylantro, Campbell, CA.
  2. Everett Anderson, BS 1998, Sun, Mountain View, CA.
  3. Shawn Atkins, BS 1998, Lucent, Columbus, OH.
  4. Matt Braun, BS 1998.
  5. Darrell Brunsch, BS 1999, Microsoft, Redmond, WA.
  6. George Edwards, BS 2004, Ph.D. student at University of Southern California.
  7. Sergio Flores-Gaitan, MS 1998, Microsoft, Redmond, WA.
  8. Priyanka Gontla, MS 2000, UBS, Irvine, CA.
  9. Pradeep Gore, MS 2000, OOMWorks, New Jersey.
  10. Tim Harrison, MS 1997, Mayasoft, Palo Alto, CA.
  11. Prashant Jain, MS 1997, IBM Research, India.
  12. Vishal Kachroo, MS 1999, Stentorsoft, CA.
  13. Michael Kircher, BS 1998, Siemens CT, Munich, Germany.
  14. Yamuna Krishnamurthy, MS 2000, OOMWorks, New Jersey.
  15. Tao Lu, MS 2003, Trading Technologies, Chicago, IL.
  16. Sumedh Mungee, MS 1998, Fujitsu, Santa Clara, CA.
  17. Bala Natarajan, MS 2000, Veritas, India.
  18. Kirthika Parameswaran, MS 2000, Telcordia, Piscataway, NJ.
  19. Stoyan Paunov, MS 2006, working at Bloomberg, NYC.
  20. Ossama Othman, MS 2002, independent consultant, Portland, OR.
  21. Marina Spivak, MS 2000, AT Desk, Charleston, SC.
  22. Nagarajan Surendran, MS 1999, Sylantro, Campbell, CA.
  23. Emre Turkay, MS 2005, Turkey.
  24. Pooja Varshneya, May 2010, Zircon Computing, Wayne, NJ.
  25. Seth Widoff, BS 1998, independent consultant, San Francisco, CA.
  26. Ming Xiong, MS 2007, currently working at AT Desk, Charleston, SC.

- *Former Staff*

  1. Chris Cleeland, OCI, St. Louis, MO.
  2. Ray Klefstad, Research Assistant Professor, University of California, Irvine.
  3. Boris Kolpackov, Independent Consultant, South Africa.
  4. Fred Kuhns, Research Associate, Washington University, St. Louis, MO.
  5. David Levine, Director of Engineering, CombineNet, Inc, Pittsburgh, PA.
  6. Will Otte, Institute for Software Integrated Systems, Nashville, TN
  7. Jeff Parsons, Optio Labs, Nashville, TN
  8. Jules White, Ph.D. 2008, Vanderbilt University, Nashville, TN

Research Support

Total research funding since June 1995: $41,899,342

- Sole PI: $12,030,403
- Co-PI: $29,868,939

Grants and Contracts Received

1. "Automated Clothing Simulation and Human Avatar Generation Engine" NSF, 9/15/2019 to 2/29/2020, $50,000.

2. "Digital Thread Modelling Environment (DTME)," AFRL (subcontract through Securboration), 8/20/2019 to 8/20/2021, $250,000, with Jules White.
3. "Creating an Evidence-based Professional Development Support Tool for Pre-K Coaches and Teachers," Department of Education (IES), $1,399,992, 7/1/18 to 6/30/22, Co-PI with Caroline Christopher.
4. "Blockchain as Middleware Services for Transactive Energy Applications," Siemens, 4/1/2017 to 9/30/2018, $274,397, co-PI Abhishek Dubey.
5. "Children Eating Well (CHEW) Smartphone Application for WIC Families," USDA 4/15/2017 to 4/14/2022, , co-PI with Pam Hull.
6. "Industrial Internet Architecture," Varian Medical Systems, Inc., 10/1/14 to 12/31/18, $288,808, co-PI Jules White.
7. Securboration, "Virtualize Combat System Environment (ViCE)," $15,000, 1/1/18-3/26/18, Co-PI with Jules White.
8. "Container Hopping at Random Intervals or Targeted-Attacked (CHARIOT)," OSD SBIR with Securboration, 1/19/17 to 1/19/18, $35,000.
9. "A Digital Platform for Social and Emotional Learning," NSF, 7/1/2018 to 12/31/2018, $50,000.
10. "Blockchains Data Exchange via FHIR," Solaster, 9/1/18 to 8/31/19, $30,000, co-PI with Jules White.
11. "Advancing Data-Driven mHealth Technologies for Long-term Health and Health Behavior Change," Trans-Institutional Program (TIPs), Vanderbilt University, 9/1/2016 to 8/31/2018, $100,000, Co-PIs Jules White, Trent Rosenbloom, and Heidi Silver.
12. "IMMoRTALS," DARPA (through subcontract with Raytheon), 12/1/15 to 12/1/19, $1,235,567, Co-PI Jules White.
13. "The Robust Software Modeling Tool (RSMT)," ONR, 7/1/14 to 6/30/17, $749,904, Co-PI Jules White.
14. "Building Resilient Distributed Systems for Next Generation Mobile Adhoc Cyber Physical Systems," Siemens 9/1/14 to 8/31/17, $438,188, co-PI Abhishek Dubey.
15. "Capability-Based Technical Reference Frameworks for Open System Architecture Implementations," OSD ASDR&E, 7/3/14 to 9/11/14, $29,690.
16. "Progressive Model Generation for Adaptive Resilient System Software," ONR STTR, 8/6/13 to 1/31/14, $49,406, co-PI Jules White.
17. "Systems and Software PRodUcibility Collaboration and Experimentation Environment (S2PRUCE2)," AFRL (subcontract through Lockheed Martin Advanced Technology Lab), 1/4/13 to 9/30/13, $108,645, with A. Gokhale.
18. "Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems," AFOSR, 10/1/13 to 9/30/16, $935,402, Co-PI(s) Aniruddha Gokhale and Xenofon Koutsoukous.
19. "Workshop on Computing Clouds for Cyber Physical Systems," NSF, 9/15/12 to 12/31/2013, $73,738.
20. "Using Social Learning to Improve Adolescent Diabetes Protocol Adherence," NIH, $1,798,029, 9/1/12-8/31/16, PI Shelagh Mulvaney.
21. "Systems and Software PRodUcibility Collaboration and Experimentation Environment (S2PRUCE2)," AFRL (subcontract through Lockheed Martin Advanced Technology Lab), 4/3/08 to 9/30/12, $381,708, with A. Gokhale.
22. "Team for Research in Ubiquitous Secure Technology (TRUST)," NSF (subcontract through UC Berkeley), 6/1/05 to 10/31/15, $5,970,900, co-PI(s) J. Sztipanovits and G. Karsai.
23. "Android Mobile Military Middleware Objects (AMMO)," DARPA, 9/30/10 to 5/02/12, $1,074,093, with S. Neema.
24. "Cyber-physical multi-Core Optimization for Resource and cachE effectS (C2ORES)", AFRL, 8/1/12 to 7/31/13, $300,000, with A. Gokhale.

25. "Model-Driven Tools for Distributed- and Multi-Core Middleware," AFRL, 4/10/12 to 10/2/12, $30,000, with A. Gokhale.
26. "Cloud Environmental Analysis and Relief," NSF, 8/1/10 to 7/31/12, $66,000, with A. Gokhale.
27. "Environment-Specific Inter-ORB Protocols," SAIC, 8/1/09 to 5/23/12, $348,350, with A. Gokhale.
28. "CoSMIC and CIAO Enhancements," Northrop Grumman, 7/1/09 to 9/30/10, $878,661
29. "Integrating DDS and CCM," Northrop Grumman, 7/1/09 to 2/15/10, $85,000
30. "Early Integration and Performance Testing of Heterogeneous Computing Environments," Australian Defence Science and Technology Organization (DSTO), 1/9/09 to 7/30/09, $180,000.
31. "Predictive Cache Modeling and Analysis," AFRL (subcontract through Lockheed Martin Aeronautics), 3/1/10 to 9/30/11, $100,000.
32. "Applications of Reliable, Fast Event Notification," Raytheon, 6/1/2008 to 5/30/2009, $60,000.
33. "Open Modular Embedded Architectures," General Electric Global Research, 8/1/2008 to 1/31/2009, $35,000.
34. "Analysis and Simulation Techniques for Next-generation Motion Control Systems," Aagard, 8/1/2008 to 1/31/2009, $13,850 with Akos Ledeczi.
35. "Open Modular Embedded Architectures," Raytheon, 8/1/2008 to 3/31/2009, $74,276.
36. "NAOMI," LMCO Advanced Technology Lab, 9/1/2007 to 11/30/2009, $290,000.
37. "IU/CRC Membership," Siemens, 1/1/2009 to 12/31/2009, $40,000.
38. "Enterprise Application Configuration in the Context of Model Driven Software Development and Software Factories," Siemens Corporate Research, 10/1/07 to 9/31/08 $91,798.
39. "Modular Extendable Demonstration of an Upgradeable Space Architecture (MEDUSA)," DARPA (subcontract through Lockheed Martin Advanced Technology Center), 2/1/2008 to 1/31/2011, $600,000.
40. "CCM Middleware Implementation and Integration," PrismTech, 6/8/2007 to 3/31/2007, $33,778.
41. "The Smart Sensor Web Architecture," NASA (subcontract through Lockheed Martin Advanced Technology Center), 12/15/06 to 11/14/09, $467,728, co-PI G. Biswas.
42. "I/UCRC Membership," General motors, 1/1/2008 to 12/31/2008, $100,000, co-PI G. Karsai.
43. "Pollux: Enhanding the Real-time QoS of the Global Information Grid," AFRL, 2/24/06 to 7/24/08, $1,242,718, co-PI M. Reiter.
44. "Intelligent Middleware for Next Generation Petascale Scientific Computing," Vanderbilt Discover Grant, 5/1/05 to 6/30/07, $100,000, co-PI(s) A. Gokhale and P. Sheldon.
45. "Air Force Center for Research on GIG/NCES Challenges," AFOSR (subcontract through UC Berkeley), 3/1/06 to 2/28/08, $600,000, co-PI J. Sztipanovits.
46. "Quality of Service Enabled Dissemination," AFRL (subcontract through BBN Technologies), 12/31/2007 to 9/30/2009, $320,000.
47. "A Fault-Tolerant Real-Time CORBA Naming Service," US Navy (subcontract through Tech-X Corp), 11/1/2007 to 4/30/2010, $175,000, co-PI A. Gokhale.
48. "System Execution Modeling Technologies for Large-scale Net-centric Systems," AFRL, 1/1/2008 to 12/31/2010, $244,000.
49. "Model-Driven Computing for Distributed Real-time Embedded Systems," Raytheon, 8/31/04 to 8/31/08, $500,000.
50. "NAOMI," LMCO Advanced Technology Lab, 9/1/2007 to 11/30/2007, $50,000.
51. "ACE/TAO Improvement Techniques and Solutions, Veritas/Symantec, 3/31/05 to 4/31/08, $198,500.
52. "Adaptive Resource Control for Certificable Systems," DARPA (subcontract through LMCO Advanced Technology Lab), 3/30/2007 to 12/31/2007, $50,000.
53. "Survivable Internet-scale Distributed Systems," IDA, 3/30/2007 to 12/31/2007, $60,000.
54. "QUality of service pICKER (QUICKER)," LMCO Advanced Technology Lab, 3/30/2007 to 12/31/2007, $60,000.

55. “Thimble,” LMCO Advanced Technology Lab, 3/30/2007 to 12/31/2007, $60,000.
56. “CADynCE Experimentation Operations (CEO),” DARPA (subcontract through LMCO Advanced Technology Lab), 8/31/2007 to 12/31/2007, $25,000.
57. “Real-time Discovery for Pub/Sub Middleware in WANs,” US Navy (subcontract through Tech-X Corp), 6/16/2007 to 9/31/2007, $15,000.
58. “GEMS Utilization Test Suite,” LMCO Advanced Technology Lab, 9/1/07 to 11/30/07, $50,000.
59. “Advanced Information Systems and Technology Program,” NASA (subcontract through LMCO Advanced Technology Center), 11/13/2007 to 12/1/2007, $22,000, co-PI G. Biswas.
60. “Design for Adaptivity and Reliable Operation of Software Intensive Systems,” NSF CNS-0613971, 9/1/06 to 8/31/08, $199,867, co-PI(s) S. Abdelwahed and G. Karsai.
61. “Software Technologies Targeting Interoperability for Systems of Systems,” Army Research Lab, 1/15/07 1/14/10, $851,567, co-PI(s) G. Karsai and J. Sztpanovits.
62. “Software Wind Tunnel (SWiT) Capabilities,” Lockheed Martin Advanced Technology Lab, 8/1/06 to 12/31/06, $60,000.
63. “High-Confidence Software Platforms for Cyber-Physical Systems,” NSF, 5/1/06 to 7/30/08, $129,179.
64. “Applying AOP to Develop of Component Synthesis with MDD,” Siemens, 3/1/03 to 2/28/07, $400,005.
65. “Addressing Domain Evolution Challenges in Model-Driven Software Product-lines,” Siemens Corporate Research, 10/1/05 9/31/07, $100,000.
66. “A Fault Tolerant Real-time CORBA Naming Service,” US Navy (subcontract through Tech-X Corp), 11/1/05 to 8/31/06, $15,000.
67. “The SYstem DEployment and Configuration AssisteR (SYDECAR),” Lockheed Martin Advanced Technology Lab, 8/1/05 to 8/1/08, $500,000.
68. “Future Combat Systems: Software Architecture Engineering,” DARPA (subcontract through Boeing), 1/28/05 to 12/31/07, $2,764,226, co-PI(s) J. Sztipanovits and G. Karsai.
69. “Development of an Eclipse Plug-in,” PrismTech, 4/28/05 to 9/30/05, $25,000.
70. “Prometheus: Enhancing the QoS of the JBI,” AFRL, 3/25/05 to 12/31/05, $500,000, co-PI(s) K. Birman and Mike Reiter.
71. “A Testbed for Assuring Quality of Software for DRE Systems,” ONR, 2/15/05 to 1/31/06, $200,000, co-PI(s) A. Gokhale and A. Porter.
72. “Enhancing the QoS of SOAs Using Eclipse-based MDD,” IBM, 2/15/05 to 1/31/06, $29,515, co-PI A. Gokhale.
73. “Model-Driven Development of BEEP Application Protocols,” Cisco, 12/15/04 to 12/14/05, $57,976, co-PI A. Gokhale.
74. “Evaluating CORBA Middleware for Space Systems,” NASA (subcontract through Lockheed Martin Advanced Technology Center), 9/23/04 to 11/30/06, $186,180, co-PI G. Biswas.
75. “Refactoring Techniques to Reduce Middleware Resource Utilization,” Qualcomm, 10/31/04 to 10/31/05, $104,000, co-P B. Natarajan.
76. “Model-Driven Development for Software Defined Radios,” BAE Systems, 12/1/04 to 3/31/05, $32,000.
77. “Enhancing the Robustness and Performance of TENA,” DISA (subcontract through SAIC and OSC), 7/1/04 to 12/31/04, $75,000.
78. “QoS-enabled Fault Tolerant Middleware and MDA Tools,” Lockheed Martin MSS, 4/1/03 to 12/31/04, $516,434.
79. “Trustworthiness in Embedded Systems,” NSF ITR CCR-032574, 9/31/03 to 8/31/06, $210,454.
80. “ACE+TAO Enhancements,” OCI, gift $20,000.
81. “Acquiring Accurate Dynamic Field Data Using Lightweight Instrumentation,” NSF ITR CCR-0312859, 10/1/02 to 9/31/07, $1,850,000, co-PI(s) A. Porter, D. Notkin, and A. Karr.
82. “Intergovernmental Personnel Act,” DARPA, 6/1/00 to 5/31/02, $198,934.

83. “Optimizing Component Models,” DARPA, 4/1/01 to 6/31/02, $210,000.
84. “HLA RTI Next-generation,” DMSO (subcontract through SAIC), 6/1/01 to 12/31/01, $70,895.
85. “ACE Enhancements for Windows NT and Windows CE,” Siemens Medical Engineering, 2/1/00 9/19/01, $112,000.
86. “Scalable and Fault Tolerant Middleware,” AFRL MURI, 12/1/99 to 3/31/02, $253,701.
87. “Protocol Engineering Research Center,” AFOSR MURI, 6/15/00 to 6/14/03, $264,720, co-PI Tatsuya Suda.
88. “Optimizing ORBs for Network Management,” Cisco Systems, 1/1/00 to 12/31/00, $100,000.
89. “TAO Optimizations,” Raytheon, 10/1/99 to 6/01/01, $50,000.
90. “ACE+TAO on pSoS,” Motorola, 8/15/99 to 12/31/99, $30,000.
91. “Real-time Distributed Object Computing,” Sprint, 8/15/99 8/14/00, $133,068.
92. “TAO Enhancements,” Krones, 8/1/99 to 9/1/99, $5,000.
93. “ACE Enhancements,” ICOMVERSE, gift, $20,000.
94. “Weapon Systems Open Architecture,” Boeing, 7/15/99 to 1/31/00, $51,491.
95. “Fault Tolerant CORBA,” Motorola Labs, 7/15/99 to 7/14/00, $139,000.
96. “TAO Enhancements,” Global MAINTECH, 7/1/99 to 8/1/99, $5,000.
97. “ACE QoS Extensions,” Motorola Trunking, 6/1/99 to 8/1/99, $5,000.
98. “CORBA Interceptors,” Experian, 5/15/99 7/14/99, $10,000.
99. “DCOM performance evaluation,” Microsoft, gift, $30,000.
100. “TAO Improvements,” OCI, 4/1/99 to 9/31/00, $27,000.
101. “Middleware Optimizations,” Telcordia, 2/1/99 to 1/31/00, $52,700.
102. “Minimum CORBA,” Hughes Data Networking, 4/1/99 to 3/31/00, $50,000, co-PI David Levine.
103. “Framework Usage Patterns,” Siemens Corporate Research, 4/1/99 to 3/31/00, $35,000.
104. “Dynamic Scheduling and Real-time ORB Optimizations,” Boeing, 10/1/98 9/30/99, $184,860.
105. “Distributed Object Computing Middleware,” Nortel, 11/1/98 10/31/99, $75,000.
106. “ACE subsetting,” “ACE subsetting,,” Nokia, 10/8/98 4/8/99, $30,000.
107. “Boeing Research Fellowship,” Boeing, 9/1/98 8/31/00, $81,486.
108. “Patterns and Frameworks Reuse Curriculum,” Lucent Bell Labs, 9/1/98 12/31/98, $31,200.
109. “Patterns, Frameworks, and Components,” Siemens ZT, 12/1/98 5/31/00, $175,000.
110. “High availability frameworks,” Lucent, 9/1/98 8/31/99, $39,400.
111. “Real-time Distributed Object Computing,” Sprint, 8/1/98 7/31/99, $288,194.
112. “Distributed Object Integration for the Quorum Project,” DARPA S30602-98-C-0187 (subcontract through BBN), 9/1/98 8/31/01, $448,643, co-PI(s) R. Schantz and J. Loyall.
113. “Evaluating a Framework for Dynamic Distributed Real-Time Scheduling,,” USENIX, gift, $18,000.
114. “Distributed Object Computing,” Microsoft, gift, $20,000.
115. “Distributed Object Visualization Environment,” Lockheed Martin, 5/1/98 to 11/31/99, $54,000.
116. “Distributed Object Computing with Adaptive End-to-end QoS Guarantees,” DARPA 9701561, 8/1/97 to 7/31/00, $873,625.
117. “Real-time CORBA for Telecommunications,” Lucent, 12/1/97 to 11/31/98, $100,000.
118. “Developing an HLA-compliant RTI with ACE,” SAIC, 12/15/97 to 1/31/00, $228,075.
119. “Real-time CORBA for Wireless,” Motorola LMPS, 10/15/97 to 10/14/98, $200,000.
120. “Real-time CORBA for Avionics,” Computing Devices International, 10/15/97 to 10/14/98, $39,050.
121. “Dynamic Scheduling of Real-time OFPs,” Boeing, 9/1/97 to 8/31/98, $224,604.
122. “Distributed Object Visualization,” Siemens MED, 10/1/97 to 9/1/98, $40,000.
123. “The ADAPTIVE Communication Environment,” Siemens MED, 10/1/97 to 9/1/98, $70,000.

124. “The Architect’s Assistant,” Siemens Corporate Research, 9/1/97 to 8/1/98, $35,000.
125. “Monitoring, Visualization, and Control of High Speed Networks,” NSF NCR-97-14698, 9/1/97 to 8/31/01, $1,200,000, co-PI(s) G. Parulkar, E. Kraemer, J. Turner, and R. Cytron .
126. “Adaptive Software Technology Demonstration (ASTD),” AFRL (subcontract through Boeing), 9/1/98 to 8/31/02, $1,200,000, co-PI(s) B. Doerr, D. Allen, and R. Jha.
127. “Patterns, Frameworks, and Components for Multimedia Systems,” Siemens Research, 1/97 to 6/98, $150,000.
128. “Adaptive Servers for High-Performance Imaging,” Kodak Networked Imaging Tech. Center, 11/96 to 11/97, $40,000.
129. “Real-time CORBA,” Sprint, 9/96 to 12/97, $345,000, co-PI G. Parulkar.
130. “OpenMAP – Object-Oriented Components for Real-time Avionics,” McDonnell Douglas, 9/96 to 9/97, $241,591.
131. “Compilation and Automatic Optimization of Network Protocol Implementations,” NSF NCR-9628218, 8/96 to 8/99, $411,025, co-PI(s) G. Varghese and R. Cytron (PI).
132. “Medical Imaging with Java and the WWW,” SIEMENS Medical Engineering, 8/96 to 7/97, $125,000.
133. “The ADAPTIVE Communication Environment,” SIEMENS Medical Engineering, 8/96 to 7/97, $90,000.
134. “High-performance Distributed Medical Imaging,” Kodak Imaging, 12/94 to 8/96, $55,152, co-PI J. Blaine.
135. “Design Patterns for Concurrent Object-Oriented Networking,” Object Technologies International, 4/96 to 4/97, $25,000.
136. “Distributed Object Computing with CORBA and DCE,” Bellcore, 5/96 to 12/96, $32,978.
137. “The ADAPTIVE Communication Environment,” SIEMENS Medical Engineering, 6/95 to 6/96, $170,000.

Courses Taught

Courses at Vanderbilt University

1. CS 215 – Intermediate Software Design, Spring 2006
2. CS 251 – Intermediate Software Design, Spring 2007, Spring 2008, Spring 2009, Fall 2009, Spring 2010, Spring 2012, Spring 2013, Spring 2014, Spring 2015, Spring 2016, Summer 2020, Summer 2021
3. CS 253 – Parallel Functional Programming with Java and Android, Fall 2020, Fall 2021
4. CS 254 – Concurrent Object-Oriented Programming with Java and Android Spring 2021
5. CS 291/242 – Software Design Studio, Fall 2004
6. CS 291/242 – Software Design Studio, Fall 2003
7. CS 292 – Beyond the Oneway Web, Fall 2008
8. CS 278 – Software Engineering, Fall 2008
9. CS 279 – Software Engineering Projects, Spring 2010
10. CS 282 – Principles of Operating Systems II, Spring 2003, Spring 2004, Fall 2005, Fall 2007, Fall 2012, Fall 2013, Fall 2014, Fall 2015, Fall 2016, Spring 2017
11. UNIV 278 – Tackling Big Questions with Mobile Cloud Computing, Fall 2016, Spring 2017, Fall 2017
12. CS 395 – Advanced Network Software Design, Fall 2006
13. CS 395 – QoS-enabled Middleware, Fall 2008
14. CS 395 – Reactive Microservices, Summer 2021
15. CS 396 – QoS-enabled Component Middleware, Spring 2005
16. CS 891 – Introduction to Concurrent and Parallel Java Programming with Android, Fall 2017

17. CS 891 – Advanced Concurrent Java Programming in Android, Spring 2018, Spring 2019, Spring 2020
18. CS 891 – Introduction to Parallel Java Programming, Fall 2018, Fall 2019
19. CS 892 – Concurrent Java Programming in Android, Spring 2017

Courses at Coursera

1. Android App Development (Android for Java; Android App Components - Intents, Activities, and Broadcast Receivers; Android App Components - Services, Local IPC, and Content Providers), 2016 to present
2. Mobile Cloud Computing with Android (Pattern-Oriented Software Architecture: Communication; Pattern-Oriented Software Architecture: Concurrency), 2014 to 2016
3. Pattern-Oriented Software Architectures for Concurrent and Networked Software, 2013

Courses at University of California, Irvine

1. ECE 011 – Computational Methods in ECE, Winter 2000
2. ECE 255 – Distributed Software Architecture Design, Spring 2000
3. ICS 142 – Compiler Theory, Summer 1989
4. ICS 23 – Data Structures, Summer 1988

Courses at Washington University, St. Louis

1. CS 562 – Advanced Object-Oriented Software Development with Patterns and Frameworks, Spring 1999
2. CS 242 – Introduction to Software Design, Spring 1998
3. CS 673 – Distributed Systems research seminar, Fall 1997
4. CS 422 – Operating Systems Organization, Fall 1997
5. CS 242 – Introduction to Software Design, Spring 1997
6. CS 544 – Distributed System Design, Fall 1996
7. Ada tasking course for McDonnell Douglas, Fall 1996
8. OO design course for McDonnell Douglas, Spring 1996
9. CS 523 – Distributed Operating Systems Organization, Spring 1995
10. CS 242 – Introduction to Software Design, Fall 1995
11. CS 673 – Distributed Systems research seminar, Spring 1995
12. CS 422 – Operating Systems Organization, Fall 1994

Other Teaching Experience

In addition to the academic teaching experience above, I have also taught numerous short-courses and tutorials on object-oriented design patterns and programming techniques, UNIX and Windows NT systems programming and network programming, C++ and C programming languages, and various distributed/networked system, compiler construction, algorithm, data structure, mobile app, and web-based cloud computing courses for the following universities and professional organizations:

- O'Reilly Live-Training
- Pearson LiveLessons
- University Extension Program, University of California, Berkeley, CA
- University Extension Program, University of California, Irvine, CA
- University Extension Program, University of California, Los Angeles, CA
- Oregon Graduate Institute of Science and Technology, Beaverton, OR
- USENIX association
- Association of Computing Machinery (ACM)
- Addison-Wesley's Technology Exchange Program, Reading, MA
- SIGS Conferences
- Object Computing Institute, St. Louis, MO
- National University, Irvine, CA

Department/School/Community Service

Service at Vanderbilt University

1. Faculty advisor for the "DataBrains" AI and Data Science student club.
2. Faculty advisor for the "Vandy Apps" student club.
3. Faculty advisor for the "BizTech" student club.
4. Led the effort to create an online Professional Masters in CS
5. Led the effort to create a continuing education program in Web Development
6. Interview panel for the Director of Professional Programs in VUSE
7. Served on the Digital Literacy committee
8. Chair of two year review committee for Taylor Johnson
9. Chair of the CS search committee in 2003, 2005, 2013, 2016, 2018
10. Chair of the committee on Big Data for the VUSE Strategic Plan
11. Member of the Provost's Special Task Force of the Data Science Visions Working Group: Trans-institutional Masters in Data Science.
12. Member of the Provost's Data Science Visions working group
13. VUSE representative for the Research IT committee
14. VUSE representative on the Provost's Digital Literacy committee
15. Reviewer for University Course proposals
16. Faculty mentor for "Accenture Garage Program"
17. VUSE representative for the Research IT committee.
18. Member of the search committee for the first Director of the Innovation Center
19. Member of the Provost's Study Group on Cross College Teaching
20. Member of the Advisory Committee for the Vanderbilt Institute for Digital Learning (VIDL)
21. Chair of the Provost's Committee on the Innovation Center
22. Member of the VUSE Career Committee
23. VUSE point of contact for VUIT
24. Committee member for Eugene Vorobeychik's promotion case to associate professor
25. Committee member for Bobby Bodenheimer's promotion case to full professor
26. Committee member for Julie Adams's promotion case to full professor
27. Committee member for Akos Ledeczi's promotion case to full professor
28. Chair of the tenure committee for Yuan Xue
29. Chair of the four year review committee for Yuan Xue
30. Member of the two year committee for Yuan Xue
31. Member of the promotion committee for Ted Bapty
32. Member of review committee for Xenofon Koutsoukos
33. Chair of promotion committee for Gabor Karsai
34. Member of promotion committee for Gautam Biswas
35. Chair of the VUSE Technology Entrepreneurship Task Force
36. Member of the VUIT faculty advisory committee
37. Owen-VUSE joint committee for 2014-2015
38. Chair of the Schmidt Family Annual Educational Technologies Lectureship
39. Member of the Provost's Study Group on Cross College Teaching
40. Chair of two year review committee for Eugene Vorobeychik

41. Member of the Chancellor's Social Media and the Internet committee
42. Member of the VU Online Education Task Force
43. Member of the ad hoc committee on EECS Industrial Advisory Board
44. Ex-officio member of the ad hoc committee on the CS graduate program
45. Ex-officio member of the ad hoc committee on the CS undergraduate program
46. Faculty facilitator for the Vanderbilt Visions program
47. Chair of the Information Technology committee for the Vanderbilt School of Engineering
48. Chair of the tenure committee for Bobby Bodenheimer
49. EECS Corporate/Internship Liaison for Computer Science and Engineering
50. Ex-officio Member of the Ad Hoc Committee on Computer Engineering
51. Faculty sponsor of the new EECS Graduate Student Organization
52. Member of the VUSE Research Institutes and Centers Council
53. Associate Chair of Computer Science and Engineering
54. Member of the Vanderbilt University Faculty Senate
55. Chair of the faculty committee on Academic Computing and Information Technology (ACIT)
56. Member of the Research Advisory Committee on Information Technology (RACIT)
57. Chair of the Systems Engineering concentration committee
58. Member of the Plan Integration and Communication Group (PICG)
59. Member of the CS graduate curriculum committee

Service at Washington University, St. Louis

1. Member of the Faculty recruiting committee
2. Member of the CS committee on recruiting industrial graduate students (RIGS)
3. Member of the CS Experimental Infrastructure for Teaching and Research (CEITR)
4. Member of the Introductory course committee
5. Member of the Graduate admission committee
6. Member of the CS representative to the CEC advisory board
7. Member of CS departmental chair search committee

Awards and Honors

1. Received the Cornelius Vanderbilt Professor of Engineering endowed chair in February 2017.
2. Received the 2015 Award for Excellence in Teaching by the Vanderbilt University School of Engineering.
3. Interviewed for Software Engineering Radio (www.se-radio.net/).
4. Vice-chair of the IEEE Chapter in middle Tennessee.
5. Elected to three year term as member of the Vanderbilt University Faculty Senate.
6. Invited speaker at the dedication of the Henry Samueli School of Engineering, along with UC Irvine Chancellor, Ralph Cicerone; Dean of the School of Engineering, Nicolaos Alexopoulos; Chairperson of the Regents of the University of California, S. Sue Johnson; President of the University of California, Dick Atkinson; and CTO and co-founder of Broadcom Henry Samueli.
7. Interviewed for Dr. Dobb's journal TechNetCast, October 24, 2000.
8. Interviewed for **iX** magazine, October, 2000.
9. Received early promotion to tenure as an Associated Professor at Washington University, St. Louis, five years after joining the faculty as an Assistant Professor in 1994.

10. Director of the "Center for Distributed Object Computing" at Washington University, St. Louis since spring of 1999.
11. Listed in Marquis' "Who's Who in Media and Communications," 1997.
12. Received joint appointment to the Mallinckrodt Institute Department of Radiology, Washington University School of Medicine, February 1996.
13. Selected to participate in the ACM OOPSLA '94 Doctoral Symposium.
14. Invited by Dr. Martina Zitterbart to participate in a 4-week international exchange program at the Universität Karlsruhe Institut für Telematik in Karlsruhe, Germany, April 1993.
15. Served as elected representative to the Associated Graduate Student organization at the University of California, Irvine from May 1991 to June 1992.
16. Served as elected graduate student representative to the Computer Science Computing Resource Committee at the University of California, Irvine from August 1988 to August 1990.

Consulting Work

1. ARINC, Fountain Valley, CA
2. ACM, NY, NY
3. Advanced Institute of Information Technology, Seoul, Korea
4. AG Communication Systems, Phoenix, AZ
5. Anderson Consulting, Chicago, IL
6. Apple, Cupertino, CA
7. AT&T Research, Murray Hill, NJ
8. BAE Systems, Greenlawn, NY
9. BAE Systems, Wayne, NJ
10. BEA, San Jose, CA
11. Bellcore, Morristown, NJ
12. BellSouth, Atlanta, GA
13. Boeing, St. Louis, MO
14. Boies, Schiller, & Flexner, Santa Monica, CA
15. Bridges & Mavrakakis, Palo Alto, CA
16. Cooley LLP, San Francisco, CA
17. Correct Care Solutions, Nashville, TN
18. Credit Suisse, Zurich, Switzerland
19. Crosskeys, Ottawa, Canada
20. DARPA, Arlington, VA
21. Desmarais, NY, NY
22. Duane Morris, Atlanta, GA
23. Edward D. Jones, St. Louis, MO
24. Envision Inc. St. Louis, MO
25. Ericsson, Cypress, CA
26. Fitzpatrick, Cella, Harper & Scinto, NY, NY
27. GaN Corporation, Huntsville, AL
28. Gibson, Dunn, & Crutcher, NY, NY
29. Goldman, Ismail, Tomaselli, Brennan, & Baum, Chicago, IL
30. Jet Propulsion Lab, Pasadena, CA

31. Kasowitz, Benson, & Torres, Redwood Shores, CA
32. Keystone Strategy, Boston, MA
33. Kilpatrick Stockton, Atlanta, GA
34. Kirkland & Ellis, San Francisco, CA
35. Kodak Imaging, Rochester, NY
36. Laureate University, Baltimore, MD
37. Lockheed Martin Tactical Systems, Minneapolis, MN
38. Lockheed Martin Mission Systems, Boulder, CO
39. Lockheed Martin Advanced Technology Lab, Cherry Hill, NJ
40. Lucent Bell Labs, Naperville, IL
41. Lucent Bell Labs, Murray Hill, NJ
42. Lucent, Whippany, NJ
43. McDonnell Douglas, St. Louis, MO
44. Microsoft, Redmond, WA
45. Morrison & Foerster, Washington DC
46. Morgan Stanley, New York, NY
47. Motorola Cellular Infrastructure Group, Arlington Heights, IL
48. Motorola Iridium, Chandler, AZ
49. Motorola Land Mobile Products, Chicago, IL
50. National Security Agency, Ft. Meade, MD
51. Naval Air Weapons Stations, China Lake, CA
52. Nortel, Ottawa, Canada
53. Object Computing Institute, St. Louis, MO
54. Object Technologies International, Ottawa, CA
55. Odetics Broadcasting, Anaheim, CA
56. Oracle, Redwood Shores, CA
57. Park, Vaughan, & Fleming, Boise, ID
58. Pearson Education, London, UK
59. Pragmatus, Alexandria VA
60. PrismTechnologies, Newcastle, UK
61. Qualcomm, San Diego, CA
62. Quinn Emanuel, NY, NY
63. Raytheon, San Diego, CA
64. Reichman Jorgensen, CA
65. Riverace, Boston, MA
66. Rubin Anders Scientific, Boston, MA
67. SAIC, Washington D.C.
68. Schwegman, Lundbert, & Woessner, Minneapolis, MN
69. Siemens Medical Engineering, Erlangen, Germany
70. Siemens Corporate Research, Princeton, NJ
71. SIGS, New York, NY
72. Software Engineering Institute, Pittsburgh, PA
73. Teradyne, Chicago, IL
74. Teledyne, Thousand Oaks, CA

75. UC Berkeley Extension, Palo Alto, CA
76. UCLA Extension, Los Angeles, CA
77. USENIX, Lake Forest, CA
78. Venable, NY, NY
79. Wong, Cabello, Lutsch, Rutherford & Brucculeri, Houston, TX
80. WMS Gaming, Chicago, IL
81. Zircon Computing, Wayne, NJ

Expert Testimony in the Past Five Years

1. March 2016, Deposed in support of Oracle in the Oracle vs. Google Fair Use trial in the United States District Court for the Northern District of California, San Francisco division. Case No. Civ. A. No. 10-03561 WHA.
2. May 2016, Testified in support of Oracle in the Oracle vs. Google Fair Use trial in the United States District Court for the Northern District of California, San Francisco division. Case No. Civ. A. No. 10-03561 WHA.
3. February 2017, Deposed in support of IBM in the IBM vs. Priceline Group case. Case No. Civ. A. N. 15-cv-137-LPS-CJB.
4. February 2018, Deposed in support of IBM in the IBM vs. Groupon case. Case No. Civ A. N. 16-122-LPS-CJB.
5. July 2018, Testified in support of IBM in the IBM vs. Groupon case. Case No. Civ A. N. 16-122-LPS-CJB.
6. August 2018, Deposed in support of Palo Alto Networks in the Palo Alto Networks vs. Implicit case. Case No. Civ 6:17-CV-182-JRG.
7. January 2019, Deposed in support of C3IoT in the E2.0 vs. C3IoT case. Case No. 1:15-cv-00530-GMS.
8. February 2019, Testified in support of C3IoT in the E2.0 vs. C3IoT case. Case No. 1:15-cv-00530-GMS.
9. June 2019, Deposed in support of IBM in the IBM vs. Expedia Inc. case. Civil Action No. IPR2018-01136.
10. July 2019, Deposed in support of Philips in the Philips vs. Microsoft case. Civil Action No. 4:18-cv-01885-HSG.
11. August 2019, Deposed in support of Philips in the Philips vs. HTC case. Civil Action No. 4:18-cv-01885-HSG.
12. August 2019, Deposed in support of Philips in the Philips vs. ASUS case. Civil Action No. 4:18-cv-01885-HSG.
13. September 2019, Deposed in support of Kroy in the Kroy vs. Groupon case. Civil Action No. IPR2019-00044.
14. September 2019, Deposed in support of Kroy in the Kroy vs. Groupon case. Civil Action No. IPR2019-00061.
15. March 2020, Deposed in support of Cisco in the Centriptal vs. Cisco case. Civil Action No. 2:18-cv-00094-HCM-LRL.
16. May 2020, Testified in support of Cisco in the Centriptal vs. Cisco case. Civil Action No. 2:18-cv-00094-HCM-LRL.

17. Jan 2021, Deposed in support of Droplets in the Droplets vs. Yahoo case. Civil Action No. 12-CV-03733-JST.

18. Jan 2021, Deposed in support of Droplets in the Droplets vs. Nordstrom case. Civil Action No. 12-CV-04049.

19. June 2021, Deposed in support of Sonos in the Sonos vs. Google case. Civil Action No. 6:20-cv-00881-ADA.

20. September 2021, Deposed in support of IBM in the IBM vs. Zillow case. Civil Action No. IPR2020-01655.

21. November 2021, Deposed in support of Apple in the Apple vs. Identity Security case. Civil Action No. 6:21-CV-460-ADA

22. January 2022, Deposed in support of IBM in the IBM vs. Chewy case. Civil Action No. 1:21-cv-01319-JSR.

Summary of Research Contributions

At Vanderbilt University I direct the Distributed Object Computing (DOC) Group at the Institute for Software Integrated Systems (ISIS), which is one of the leading research groups in the world on middleware platforms and MDE tools for DRE systems and mobile cloud computing platforms. Over the past several decades I have conducted and managed research projects on a range of topics, including patterns, optimization techniques, and empirical analyses of software frameworks that facilitate the development of quality of service (QoS)-enabled middleware and model-driven engineering (MDE) techniques/tools for distributed real-time and embedded (DRE) systems and mobile cloud computing apps running over wired/wireless networks and embedded system interconnects. The research methodology throughout my career has involved:

- *Creating* innovative middleware and MDE technologies technologies, such as design formalisms, QoS specification/enforcement techniques, end-to-end and cross-layer middleware optimizations, and automated tools for specifying, analyzing, and synthesizing dependable DRE software from higher-level domain-specific models.
- *Applying* these technologies in conjunction with colleagues in academia and industry to demonstrate and mature middleware and MDE technologies and tools in the context of production mission-critical DRE systems.
- *Amplifying* the adoption and transition of these technologies in both academia and industry via 625+ technical papers, 575+ tutorials and invited talks, millions of lines of popular open-source software, and scores of innovative face-to-face and online courses published and delivered to more than 300,000 students around the world.

The R&D efforts I have led have had a significant impact on academic research and commercial practice. For example, dozens of universities throughout the world use the middleware and MDE tools my DOC Group has developed as the basis for their research and teaching efforts. Moreover, the open-source middleware frameworks and MDE tools generated from projects I've led constitute some of the most successful examples of software R&D ever transitioned from research to industry, being widely used by thousands of companies and agencies worldwide in many domains for three decades. For example, the ACE and TAO middleware frameworks developed by the DOC Group are used by developers in thousands of companies (such as Boeing, Cisco, Ericsson, Kodak, Lockheed Martin, Lucent, Motorola, NASA/JPL, Nokia, Nortel, Raytheon, SAIC, Siemens, Sprint, and Telcordia) in a wide range of domains (such as telecom/datacom, healthcare, process automation, avionics, homeland security and defense, financial services, online gaming, social media, and distributed interactive simulation).

Teaching Contributions and Impact

I have taught scores of cutting-edge courses on topics relating to object-oriented design and programming, software patterns, middleware for distributed real-time and embedded systems, concurrent and networked programming with C++ and Java, and mobile cloud computing with Android. I received the 2015 Award for Excellence in Teaching by the Vanderbilt University School of Engineering. In addition, I've taught

10 popular MOOCs at Vanderbilt on topics related to pattern-oriented mobile cloud computing with Android to over 200,000 learners from around the world.

I recently created and co-taught one of the first cross-college University Courses at Vanderbilt on "Tackling Big Problems with Mobile Cloud Computing," where ten highly diverse teams consisting of 11 arts and science students and 44 computer science students were mentored by 11 faculty from the College of Arts and Sciences, the School of Nursing, the School of Law, the School of Medicine, the School of Engineering and Vanderbilt University Medical Center. The projects in this course addressed relevant, real-world problems involving mobile cloud computing technologies, including:

- Effectively engaging young people with chronic diseases and medical conditions, such as diabetes, asthma and obesity
- Creating "smarter" cities and sustainable energy platforms via an app-based transportation hub for Nashville, and remotely monitoring the safety and operations of novel sources of power, including solar, wind and natural gas, and
- Helping economically disadvantaged individuals bridge the digital divide to obtain better guidance on medical and legal matters.

Summary of Career Accomplishments

My career accomplishments include the following:

Publications and presentations. I have published 650+ works (127 journal papers, 195 conference papers, 5 books, 4 book-length reports, 3 edited book collections, 64 book chapters, 74 workshop papers, 13 short papers and posters, 75 trade magazine columns/articles, and 101 editorials and book forewords). My papers have appeared in the most selective journals (*e.g.*, ACM Transactions in Embedded Computing Systems, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Software Engineering, IEEE Transactions on Computing, IEEE Journal of Selected Areas of Communications, and ACM Transactions on Autonomous and Adaptive Systems) and conferences (*e.g.*, ACM SIGCOMM, ACM OOPSLA, IEEE INFOCOM, IEEE ICDCS, IEEE RTAS, ACM/IEEE Middleware, and the ACM/IEEE ICSE) in my field. I have also given 600+ invited lectures and tutorials world-wide.

Measures of scholarly impact. My publications have been cited 45,000+ times across a comprehensive spectrum of high-impact venues. My h-index is 87 and my i10 index is 401. These bibliometrics indicate the significant impact of my publications as a researcher in the field of Computing.

Funding. Since June 1995 I have been a PI or co-PI for grants, contracts, and gifts totaling more than $41 million dollars. I have been the sole PI for over $11.5 million dollars of this amount.

Graduate advising and training. During my academic career I have (co-)advised and graduated 19 doctoral students and over 25 masters students.

Professional service and leadership. I have engaged in the following professional service and leadership capacities during my career:

- Served as guest editor of 12 ACM, IEEE, and USENIX journals, and served as editor-in-chief of the C++ Report magazine.
- Served as general chair or program (co)-chair for 35 conferences, tutorial chair for 4 conferences, co-organized 14 workshops, and served on the program committees for over 245 ACM, IEEE, IFIP, USENIX, and OMG conferences.
- From 2013 to 2015 I served on the Advisory Board for the joint US Navy/Army Future Airborne Capability Environment (FACE).
- From 2013 to 2015 I served as co-lead of a task area on "Published Open Interfaces and Standards" for the US Navy's Open Systems Architecture initiative.
- From 2010 to 2014 I served a member of the Air Force Scientific Advisory Board, where I was the Vice Chair of a study on Cyber Situational Awareness for Air Force mission operations.
- From 2006 to 2011 I served as the Chief Technology Officer for the Software Engineering Institute at Carnegie Mellon University (2010 to 2011), Zircon Computing (2009 to 2010), and Prism Technologies (2006-2008), where I was responsible for directing the technical vision and strategic R&D investments.
- From 2000 to 2003 I served as a Program Manager at the DARPA Information Technology Office (ITO) and Information eXploitation Office (IXO) the Deputy Director for DARPA ITO, where I lead the national R&D effort on QoS-enabled middleware for DRE systems.

- From 2001 to 2003 I served as Co-chair for the Software Design and Productivity (SDP) Coordinating Group, which formulates the multi-agency research agenda in fundamental software design for the Federal government's Information Technology Research and Development (IT R&D) Program, which is the collaborative IT research effort of the major Federal science and technology agencies.

University service and leadership. I have engaged in the following service and leadership capacities at Vanderbilt University during the past two decades:

- **Associate Provost of Research**. I became the Associate Provost for Research at Vanderbilt University in July of 2018. In this capacity I am responsible for developing cohesive and sustainable information technology (IT) services to advance research and scholarship across Vanderbilt's ten schools and colleges, including scalable and secure storage, processing, and communication solutions; big data research cores and corerelated services, and NIST 800-171 compliant IT services. I am also responsible for overseeing Vanderbilt's new "liquid workforce" service that provides researchers with on-demand access to shared technology expertise to help them develop research IT solutions, especially with data-intensive workflows, while also enabling shared software developers to add value to multiple research programs throughout the university.
- **Data Sciences Initiatives**. I am deeply involved in Vanderbilt's initiatives on Data Science. Starting in August 2018, I became a founding Co-Director of the Data Science Institute at Vanderbilt. During the past year I also chaired the ad hoc committee on Big Data for the Vanderbilt University School of Engineering (VUSE) strategic planning process, as well as served on the Provost's Special Task Force on a trans-institutional Masters in Data Science and the Provost's Working Group on Data Science Visions, which sets the direction for trans-institutional Data Science research. I also created and led a presentation on "Big Data" for the Vanderbilt University Board of Trust in the spring of 2017 that helped initiate Vanderbilt's investment in the Data Science Institute.
- **Cross-College Teaching**. I am a leader in Vanderbilt University's forays into Cross-College teaching. For example, I served as a member of the Provost's Study Group on Cross College Teaching, which formulated the concept of "University Courses" that brings faculty together from multiple schools to actively engage students of diverse backgrounds and promote new and creative trans-institutional learning. I also created/taught one of the first University Courses on "Tackling Big Problems with Mobile Cloud Computing." Each semester since the fall of 2016 I've taught this course in a multidisciplinary environment where undergraduate and graduate students from multiple schools team with computer science students to address big questions, such as how mobile cloud computing technologies can engage young people with chronic diseases; change political discourse in the United States and around the world; and help economically disadvantaged individuals bridge the digital divide to obtain better guidance on nutrition and legal matters. I also spearheaded the effort to create a CS 1000 course on "the beauty and joy of computing" that is intended for non-CS majors at Vanderbilt University.
- **Digital Learning**. I play a significant role in Vanderbilt's digital learning initiatives, including teaching (1) the first Massive Open Online Course (MOOC) at Vanderbilt in 2013 on "Pattern-Oriented Software Architecture for Concurrent and Networked Systems," (2) the first trans-institutional MOOC Specialization (together with the University of Maryland, College Park) in 2014 on "Mobile Cloud Computing with Android," (3) a Coursera Specialization on "Android App Development" since the spring of 2016, and (4) the forthcoming online Computer Science professional master's degree being created in conjunction with 2U. I have also played a key role in formulating the Vanderbilt digital learning strategy as a member of the Advisory Committee for the Vanderbilt Institute for Digital Learning (VIDL), a member of the Vanderbilt Online Education Task Force, a member of the Chancellor's Social Media and the Internet committee, chair of the Schmidt Family Annual Educational Technologies Lectureship, and a member of the Provost's committee on Digital Literacy whose charter is to ensure that all Vanderbilt students learn computational thinking in their undergraduate experience.
- **Technology Entrepreneurship**. I have been highly engaged in entrepreneurship leadership at Vanderbilt over the past five years. In particular, I chaired the VUSE Technology Entrepreneurship Task Force and the Provost's Committee on the Vanderbilt Innovation Center, known as the Wond'ry (I also served as a member of the search committee for the first Director of the Wond'ry Innovation Center). I am one of the inaugural faculty mentors for the "Garage Program at the Wond'ry, where I mentor multi-disciplinary teams of undergraduate and graduate students to help

companies (such as Accenture and RGP) establish new lines of business, e.g., liquid workforce services for the oil and gas domain, supply chain risk management using blockchain technologies, etc. I also serve as the faculty advisor for the VandyApp, DataBrains, and BizTech student organizations, which teach software development skills, prepare students for technical job interviews, and foster a welcoming and diverse environment for high-tech entrepreneurship collaboration across campus.

- **EECS Department Leadership**. I served as the Associate Chair of the Electrical Engineering and Computer Science (EECS) department at Vanderbilt University from 2004 to July 2018. In this capacity I worked with the EECS Chair to provide intellectual leadership and assist in EE, CS, and CompE faculty hiring, curricular development, and course staffing. I also represented Vanderbilt at the bi-annual CRA "CS Chairs" meeting at Snowbird Utah since 2008. In the past several years I focused on innovative digital learning techniques (such as pre-recording material and/or recording lectures in class so students can listen/watch to them at their leisure to ensure they master the course material) to handle the surge in undergraduate CS enrollment without adversely affecting Vanderbilt's commitment to high quality education. I also spearheaded several initiatives to create a continuing education program focused on web development in partnership with Trilogy Education Services and a professional masters degree program in CS in conjunction with 2U.
- **Information Technology Infrastructure for Research**. Over the past two decades I've played a leadership role in the Vanderbilt University Information Technology (VUIT) planning and governance processes. In addition to my latest role as the Associate Provost for Research, I've also chaired the faculty committee on Academic Computing and Information Technology (ACIT), served as the VUSE point of contact for VUIT, the VUSE representative for the Research IT committee as a member of the VUIT faculty advisory committee, as well as served as a member of the Research Advisory Committee on Information Technology (RACIT), and a member of the Provost's Research IT Special Project Working Group, which focuses on supporting the research needs of all schools at Vanderbilt.

EXHIBIT B

MATERIALS RELIED UPON

DEPOSITIONS

Deposition of David Kleidermacher (February 3-4, 2022)

Deposition of Lawrence Koh (December 9, 2021)

Deposition of Tian Lim (December 2, 2021)

BATES NUMBERED DOCUMENTS

EPIC_GOOGLE_01761929
EPIC_GOOGLE_03982547
GOOG-PLAY-000005203.R
GOOG-PLAY-000042623.R
GOOG-PLAY-000046830.R
GOOG-PLAY-000219435.R
GOOG-PLAY-000297605.R
GOOG-PLAY-000343365.R
GOOG-PLAY-000398862
GOOG-PLAY-000558461.R
GOOG-PLAY-000575018.R
GOOG-PLAY-000880576.R
GOOG-PLAY-001076451
GOOG-PLAY-001076452.R
GOOG-PLAY-001088593
GOOG-PLAY-001285448
GOOG-PLAY-001317740
GOOG-PLAY-002011285.R
GOOG-PLAY-002910052.R
GOOG-PLAY-004268238.R

GOOG-PLAY-004284846

GOOG-PLAY-004494298.R

GOOG-PLAY-004506533

GOOG-PLAY-004704453.R

GOOG-PLAY-004722290

GOOG-PLAY-004904016.R

GOOG-PLAY-005563726

GOOG-PLAY-005649820

GOOG-PLAY-005653611

GOOG-PLAY-005653612.R

GOOG-PLAY-005659061

GOOG-PLAY-006997722.C

GOOG-PLAY-007271749

GOOG-PLAY-007271750

PUBLICLY AVAILABLE MATERIALS

Google I/O 2013 – What's New in Google Play Services = https://www.youtube.com/watch?v=49pWckcaZEI at 3:34

Android – Developers, Platform Architecture *available at* https://developer.android.com/guide/platform

Android – Developers, Device compatibility overview, *available at* https://developer.android.com/guide/practices/compatibility

Android – Source, Android Compatibility Program Overview, *available at* https://source.android.com/compatibility/overview.html

https://developers.google.com/location-context/fused-location-provider

https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/

https://material.io/develop/android

Android – Source, Android 12 Compatibility Definition, *available at* https://source.android.com/compatibility/12/android-12-cdd

https://www.xda-developers.com/android-12-alternative-app-stores-update-apps-background/

https://support.google.com/googleplay/android-developer/answer/9858738

https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works

https://developer.android.com/studio/publish/app-signing

https://www.paypal.com/us/webapps/mpp/paypal-checkout

https://www.braintreepayments.com

https://stripe.com

https://www.authorize.net

https://www.shopify.com

https://www.shopify.com/partners/blog/shopify-android-buy-sdk

https://paymentservices.amazon.com/docs/EN/23c.html

https://firebase.google.com/products/extensions/stripe-firestore-stripe-payments

https://paddle.com/platform/in-app-purchase

https://stripe.dev/stripe-android/

https://github.com/paypal/android-checkout-sdk

https://developer.authorize.net/api/reference/features/in-app.html

REDACTED VERSION

Exhibit A5
to
C. Cramer Declaration

HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

IN RE GOOGLE PLAY STORE ANTITRUST LITIGATION THIS DOCUMENT RELATES TO: *In re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD *In re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD	Case No. 3:21-md-02981-JD

EXPERT REPORT OF DR. MICHELLE M. BURTIS

March 31, 2022

TABLE OF CONTENTS

HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY

List of Tables

List of Figures

I. INTRODUCTION AND ASSIGNMENT

1. I am a Senior Consultant at Charles River Associates, an economic and finance consulting firm with offices in the United States as well as internationally. I have a Ph.D. in Economics from the University of Texas at Austin and have published in the field of economics.

2. In my work, I have studied and analyzed various forms of business conduct and how that conduct may affect the performance of markets and individual firms. I have analyzed such business conduct in antitrust cases, in other forms of commercial litigation, and in government regulatory proceedings. My analyses have included instances of conduct related to conspiracies, such as price fixing and bid-rigging, tying, monopolization, and monopsonization. I have also analyzed issues related to whether certain antitrust allegations, such as price-fixing and tying, can be addressed with common evidence or must be addressed through individualized inquiry and have published articles related to those issues. I have submitted testimony in the courts and in private arbitrations. I have taught undergraduate microeconomics at the University of Texas and graduate economics at George Mason University. A copy of my curriculum vitae, including a list of matters in which I have testified as an expert in the past five years, is included as Appendix A. My normal and customary billing rate is $900 per hour. My compensation is not contingent or based on the content of my opinions or the outcome of this matter.

3. I have been asked by counsel for the Defendants Google LLC, Google Ireland Ltd., Google Commerce Ltd., Google Asia Pacific Pte. Ltd and Google Payment Corp. (collectively, "Google") to review and respond to the opinions offered in the expert reports of Dr. David S. Sibley,[1] Dr. Michael A. Williams,[2] and Dr. Hal Singer,[3] in which they assert that Plaintiffs can determine the fact of antitrust injury and measure alleged damages using common evidence and common methodology for all proposed class members, as well as to address economic considerations regarding whether doing so is possible.

4. To prepare this report I have relied on the materials and sources listed in the Appendix B.

5. The remainder of this report is organized as follows. Section II includes a summary of my opinions regarding Developer Plaintiffs' and Consumer Plaintiffs' claims of class wide impact. Section III summarizes assumptions. Section IV provides background information about the products at issue (e.g., apps, subscriptions, and IAPs), Google Play, and proposed class members. Section V describes the economic framework I used to analyze Plaintiffs' claims of classwide antitrust impact. Section VI describes the economic reasons and evidence contradicting Consumer Plaintiffs' and Developer Plaintiffs' claims of classwide impact. Section VII discusses the Developer Plaintiffs' experts' opinions and analyses and describes why they fail to prove classwide impact using common evidence. Section VIII discusses the Consumer Plaintiffs' expert's opinions and analyses and describes why they fail

---

1 Expert Report of Dr. David. S. Sibley, February 28, 2022, *In Re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD ("Sibley Report").

2 Expert Report of Dr. Michael A. Williams, February 28, 2022, *In Re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD ("Williams Report").

3 Expert Report of Dr. Hal J. Singer, February 28, 2022, *In Re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD and Case No. 3:21-md-02981-JD ("Singer Report").

to prove classwide impact using common evidence. Section IX discusses Plaintiffs' experts' opinions regarding Google Play Points. Section X concludes.

II. SUMMARY OF OPINIONS

6. Two proposed classes – a proposed developer class and a proposed consumer class –allege that they were impacted by Google Play's alleged anticompetitive conduct in certain alleged relevant antitrust markets.

7. The primary issue addressed in this report is whether all members of each of the two proposed classes were negatively impacted by Google's alleged conduct ("classwide impact") and whether classwide impact can be proven with common evidence. For the reasons explained below, classwide impact cannot be established using predominantly common evidence for either the proposed class of developers or the proposed class of consumers.

8. Developer Plaintiffs claim to represent a class of U.S. Developers that paid a service fee of greater than 15% on the sale of apps, subscriptions, or in-app products ("IAPs") in the Google Play store ("Google Play").[4] Developer Plaintiffs claim that absent the alleged conduct, Google Play's service fees would have been lower and that developer class members' injury is equal to the difference between the service fees they paid in the actual world and the allegedly lower service fees they would have paid in the "but-for" world.

9. Consumer Plaintiffs claim to represent a class of consumers in the U.S. who made purchases of apps, subscriptions, and IAPs in Google Play. Consumer Plaintiffs claim that the alleged anticompetitive conduct led to higher service fees and that those higher fees were largely passed through to them in higher prices for app, subscription, and IAPs. Consumer Plaintiffs' experts alternatively assert that, in a world where Google faced more competition, Google would have provided consumers with more "loyalty points" in Google's Play Points program.

10. As an initial matter, individualized analysis is needed to identify members of the putative developer class. Developer Plaintiffs define a class of "U.S." developers (including additional criteria such having apps with payments, rather than free apps), but have not stated what constitutes a "U.S." developer in their proposed class definition. Developer Plaintiffs' expert identifies members of the proposed developer class using a "country-code" in a certain Google dataset. But the country codes in that Google dataset are not always consistent with other Google datasets. Moreover, the use of the country-code in the dataset indicates some developers are "U.S." developers when the developers appear to be foreign companies.[5] Developer Plaintiffs have no way, without engaging in an individualized inquiry into the circumstances of each developer in the database, to identify which members of their proposed class are "U.S." developers.

---

[4] Second Amended Consolidated Class Action Complaint for Violation of the Sherman and Clayton Acts (15 U.S.C. ¶¶ 1, 2, 3, 15, 26), Cartwright Act (Cal. Bus. & Prof. Code ¶¶ 16700 et seq.) and Unfair Competition Law (Cal. Bus. & Prof. Code ¶¶17200 et seq.), January 21, 2022 ("Developer Complaint") at ¶244.

[5] See Appendix C for details.

A. Developer Plaintiffs' and Consumer Plaintiffs' Claims of Classwide Antitrust Impact Depend on Showing the Service Fee Rates for All Developers and Apps Would be Lower in the But-For World

11. Developer Plaintiffs' and Consumer Plaintiffs' claims of classwide antitrust impact depend on showing that there would be a reduction in service fees for all developers. Consumer Plaintiffs assert that the service fee reduction would be passed on in lower prices for apps, subscriptions, and IAPs. Developer Plaintiffs assert that the reduction generally would not be passed on to consumers in the form of lower prices. Developer Plaintiffs experts claim that service fee rates would not only be lower in the but-for world but would be uniform across all apps and developers. Consumer Plaintiffs expert offers two alternatives: either there would be one service fee rate reduction for paid apps and a different rate reduction for subscriptions and IAPs, or there would be a single rate reduction for all transactions, apps, and developers.

12. Economic evidence demonstrates that not all service fee rates would fall or be uniform in Plaintiffs' but-for world and that determining which developers and which apps would obtain lower rates requires an individualized analysis of each developer and app.

13. [redacted]

14. Indeed, when Google Play and other app stores have reduced service fee rates, they have generally done so only for certain apps or developers. Targeted reduction of service fee rates by app stores reflects that some developers can negotiate lower service fees because they have popular apps that are valuable to Google Play and/or have business models or products that make them particularly well-suited to make use of other distribution or monetization options that do not involve payment of fees to Google. In the rare instances when an app store other than Google Play provided a lower service fee rate to all developers, the app stores' rivals, including but not limited to Google Play, did not respond in kind.

15. Thus, economic factors explain why competition in the actual world does not result in "across-the-board" rate reductions to all developers or for all apps. Plaintiffs' experts must address – but do not – that the economic factors explaining these dynamics in the actual world would also exist in the but-for world. Those characteristics include the unique circumstances of developers in terms of their business strategy and the relative popularity of their apps, which would continue to exist in the but-for world. Therefore, even assuming an increase in competition in a but-for world, some developers likely would be able to obtain lower rates or more services from app stores, but other developers would not.

16. Even assuming Plaintiffs' contention that in a but-for world, developers could use other billing systems and thereby avoid paying Google any service fees, some developers likely would not have a less expensive option for payment processing. Many payment processors charge a flat fee and a percentage fee for each transaction. For developers that set subscription or IAP prices at low price points – such as $0.99 – a flat fee such as $0.49

charged by PayPal plus a percentage fee add up to over 52% of the transaction price, far above Google Play's 30% service fee which covers not just payment processing, but additional services as well. This would be the case for the 22% of putative developer class members that sell apps, subscriptions, or IAPs *only* at prices of $0.99. Since those putative developer class members likely would not have a lower priced alternative for payment processing in a but-for world, their service fees would likely remain unchanged and therefore the prices that they set for the members of the proposed consumer class would remain unchanged as well. Developer Plaintiffs' experts do not address this issue at all. Consumer Plaintiffs' expert assumes, without any basis, that alternative payment processors' flat fees would not exist in the but-for world. If these 22% of putative developer class members would not have a less expensive payment processing option in the but-for world, then they are uninjured. Consumers who use those developers' apps also could not show that they would have paid a lower price in the but-for world and therefore are not injured.

17. Neither Developer Plaintiffs' experts nor Consumer Plaintiffs' expert provide an economic basis for their assumption that all developers' and all apps' service fee rates would be lower (and uniform) in the but-for world.

18. Developer Plaintiffs' expert, Dr. Sibley, assumes that in the but-for world an increase in competition would lead to lower rates for all developers. The assumption is contradicted by Dr. Sibley's own observation that in the actual world, certain developers took advantage of their ability to use other options for app distribution in order to – in his words – "resist" Google's alleged anticompetitive conduct and obtain competitive rates. Dr. Williams, another of Developer Plaintiffs' experts, adopts those actual-world rates as a competitive benchmark.

19. Dr. Sibley's other rationales for assuming all developers' rates would be lower in the but-for world are equally baseless. For example, he relies on an inaccurate and incomplete narrative about service fee rate changes among PC app stores. He claims that when the Epic Game Store entered with a lower rate, the Microsoft Store reduced its rate. But he fails to acknowledge that Microsoft's rate change came over two years after Epic Game Store's entry and fails to consider that another large PC app store, Steam, reduced rates for certain developers but not others. His claim that Google would not (or could not) set different service fees is contradicted by the fact that Google Play already sets rates differently for different developers, and that Google individually negotiates terms for certain developers. Dr. Sibley's assertion of uniformly lower service fee rates in the but-for world is at odds with economic theory showing that setting different prices can be optimal for firms and at odds with the reality that many firms in competitive industries utilize differential and increasingly sophisticated and complex pricing systems.

20. Consumer Plaintiffs' expert, Dr. Singer, similarly presents no economic basis or evidence to justify his assumption that service fee rates for all paid apps, all subscriptions, and all IAP would be lower in the but-for world. Dr. Singer's claim about lower service fee rates is based on mathematical calculations that simply assume that in the but-for world, an increase in competition will have the same effect on *all* apps and developers. By design, each of his calculations can return only a single but-for service fee rate – he allows for no other possibility. Thus, Dr. Singer has assumed that all service fee rates would be lower. He has not proposed a method for proving why this would be so for all or virtually all members of the putative consumer class.

B. Developer Plaintiffs' Claims Conflict with Consumer Plaintiffs' Claims of Antitrust Impact on Retail App, Subscription, and IAP Prices

21. Consumer Plaintiffs and Developer Plaintiffs have conflicting claims about antitrust impact as it relates to the prices of apps, subscriptions, and IAPs in the but-for world. Consumer Plaintiffs claim they were impacted by the alleged conduct because virtually all developers would have largely passed through lower service fee rates in the form of lower prices. Conversely, Developer Plaintiffs claim that very few prices would be lower in a but-for world with lower service fees.

22. Given the two sets of Plaintiffs' contradictory claims regarding prices and antitrust impact, a substantial portion of the two proposed classes' damages claims are duplicative of each other. Developer Plaintiffs claim impact and damages stemming from the difference between actual and but-for service fees, which they claim they would retain rather than pass through in the form of lower prices to consumers. Consumer Plaintiffs claim broadly the same impact and damages on the basis that developers would largely pass through any difference in service fees to them in the form of lower prices. As a result, the two proposed classes' damages claims double count much of the alleged overcharge.[6]

23. The economic evidence shows that the existence (or lack thereof) of pass-through is not a common issue for Consumer Plaintiffs or Developer Plaintiffs. Rather, my analysis shows that some but not all developers would pass through lower service fee rates to consumers and determining which prices would be lower requires an individualized app-by-app analysis. Although economic models can be used to identify certain variables that inform the issue of which prices are likely to be lower in the but-for world, those models are based on assumptions about the way developers set prices. In particular, the models assume that every developer will set a price to maximize short-term profits rather than longer-term goals, such as attempting to maximize the number of consumer downloads with low prices and then raising prices. Those assumptions do not hold for all developers but vary from developer to developer.

24. Furthermore, in economic models that assume short-term profit-maximization there are important supply and demand variables that determine whether a given developer will reduce prices when a service fee rate changes. These variables – for example, the marginal cost of apps and the price elasticity of demand for apps – vary across apps. Determining these variables for an app requires individualized analysis. For that reason, determining whether a service fee rate reduction would be passed through to consumers in the form of lower prices

---

6 Over the period August 2016 to December 2020 (a period common to both Plaintiffs' damage claims), Developer Plaintiffs' damage estimates range from \$2.5 billion to \$2.6 billion. Of those amounts, roughly 54% (\$1.3 billion to \$1.4 billion) are related to sales to members of the putative consumer class. Over the same period, Consumer Plaintiffs' damage estimates range from \$2.4 billion to \$4.7 billion. Of those amounts, roughly 35% (\$0.8 billion to \$1.7 billion) are related to purchases from members of the putative developer class. Given the two sets of Plaintiffs' contradictory claims regarding prices and antitrust impact, these Consumer Plaintiffs' damages related to purchases from Developer Plaintiffs are duplicative of the Developer Plaintiffs' damages related to sales to U.S. consumers. See this report's production.

also requires individualized analysis. Those assumptions do not hold for all developers but vary from developer to developer.

25. Developers' price-setting strategies are also highly relevant to the issue of pass-through. Many app developers set prices to end in "99" (e.g., \$0.99, \$1.99). Even if such a developer sets prices to maximize short-term profits, it does so subject to the constraint that it will change a price only if the change from one price that ends in "99" is more profitable than another price that ends in "99." This factor limits the extent to which developers will be inclined to reduce prices in the but-for world even if they obtain a lower service fee rate ("sticky prices").

26. The relevance of these economic factors to pass-through is confirmed by empirical evidence. The data produced by Google show that prices changed only for some of the apps, subscriptions, and IAPs that had a service fee rate change. Indeed, many of these apps, subscription, and IAP prices never change throughout the period for which they appear in the transaction data. That empirical evidence is consistent with the factors described above. Many apps are likely to have low (or close to zero) marginal costs and many developers' pricing strategies have an inherent stickiness.

27. Developer Plaintiffs' expert, Dr. Williams, analyzes some of the available Google data and similarly finds that few prices fall after a service fee rate change. But Dr. Williams' conclusion based on that data – that "pass-through of lower service fees is minimal in this market" – is overly broad, and he does not consider any of the economic factors that inform whether prices would be lower with lower service fee rates. Dr. Williams does not attempt to analyze the marginal costs of any app, the demand elasticity of any app, or the pricing strategy of any developer.

28. Consumer Plaintiffs' expert, Dr. Singer, adopts a different method for evaluating pass-through – one that generates demonstrably false results. Dr. Singer does not consider whether actual service fee rate changes have led to price changes in the actual world. Instead, he calculates a pass-through rate based on a simplistic formula derived from a certain model of demand: one minus the developer's share of sales of an app category. For example, if Dr. Singer identified a developer that offered an app in Google Play's Games category and calculated its share of Game sales as 1%, Dr. Singer would conclude that the pass-through rate for that game was 99% (i.e., (1- 0.01 = 0.99) or 99%). This calculation does not include service fees or prices for this app. Whether service fees fell for this app and led to lower prices – or did not – is not part of Dr. Singer's calculation. In this way, Dr. Singer finds about 95% of his pass-through rates are close to 100%; that is, nearly complete pass-through. Indeed, Dr. Singer's "method" would find pass-through for every developer unless that developer had a 100% share in its app category. Given that Dr. Singer used very broad app categories, with thousands of apps in many categories, where a single app does not have a 100% share, his method *guarantees* nearly universal positive pass-through rather than proves that it would occur for all apps.

29. Dr. Singer's method produces results that are verifiably wrong. Dr. Singer's method "finds" nearly complete pass-through for apps, subscriptions, and IAPs that had a service fee rate change in the actual world but whose prices did not change *at all*. Dr. Singer could have calculated pass-through rates from the actual data to determine whether his formula-based method produced reliable results to be used for a but-for world, but he did not.

30. Dr. Singer did not analyze any of the economic factors that inform the issue of pass-through. He did not attempt to estimate or investigate the cost conditions for any app (even though he sets out the theoretical result that there cannot be any pass-through of a service fee rate change if marginal cost is zero). He did not consider developers' pricing strategies, or the use of prices that end in "99." And he made unrealistic and unsupported assumptions about which apps are substitutes for each other. Instead, he relied on a simplistic formula that produces results that are wrong. His method is not capable of identifying those apps for which a service fee rate change leads to a price change, or the amount of any such pass-through.

31. Importantly, Dr. Singer's failure to offer a reliable method to determine pass-through rates invalidates his calculations related to but-for service fee rates, as well. Dr. Singer uses the average of his calculated pass-through rates to find what service fees would have been in a but-for world. Because his pass-through rates are based on a flawed methodology, his average pass-through rate is wrong, and his but-for service fee rates are necessarily flawed as well.

C. Developer Plaintiffs' Focus on Claimed Overcharges Ignores Critical Factors That Must Be Analyzed to Determine Antitrust Impact

32. Developer Plaintiffs' expert Dr. Williams contends that analyzing a developer's lost profits is not necessary to measure impact or damages because it would only show "additional" lost profits that would have been generated for a developer that passed through some (or all) of the service fee rate changes. This opinion is flawed for several reasons.

33. Dr. Williams' conclusion is based on the notion that the pass-through rates he has found are representative of *all* pass-through rates, even though he has done no study of any of the economic variables that are relevant to pass-through.

34. Moreover, developers' costs cannot be ignored in a lost profits analysis. Materials produced in this case, including deposition testimony from Epic Games and developer class representatives, show that at least for some developers, more app stores would increase developers' costs. Dr. Williams ignores these facts and assumes that no developer's costs would be higher in the but-for world.

35. Dr. Williams further ignores that Google would have a strong economic incentive to change the way it monetizes Google Play in Plaintiffs' but-for world. Depending on how Google would change its monetization strategy, different developers could pay higher or lower fees in the but-for world. For example, [redacted]. For some developers, this [redacted] would mean higher costs. Peekya, one of the developer class representatives had total fees of $28.60 over the period it offered its app in Google Play (including the one-time $25 developer fee). If Google Play had charged [redacted]. Peekya would have paid [redacted] in fees. 67% of developers paid less than [redacted] in service fees, so those developers would pay the same amount of fees, or more, if Google Play changed its monetization to [redacted] in the but-for world. Dr. Williams overlooks evidence that Google has considered multiple alternative monetization strategies and that depending on the strategy, some developers' fees would be higher in the but-for world.

D. Additional Reasons Consumer Plaintiffs Cannot Show Common Proof of Classwide Antitrust Impact

36. Dr. Singer claims in the alternative that in the but-for world Google might expand its Google Play Points loyalty program, instead of reducing service fees. However, Dr. Singer's Play Points model cannot be used to prove classwide impact. Dr. Singer fails to consider that in the actual world, only about [redacted] of U.S. consumers participated in and have redeemed Play Points. Given that low redemption rates in loyalty programs are not unusual (and are not the result of any alleged conduct), there is no basis to assume that all U.S. consumers would participate in an expanded program and therefore no basis to assume that all consumers would be better off in this version of Plaintiffs' but-for world.

37. Dr. Singer also does not consider that, in a but-for world, Google could change its monetization strategy and charge service fees for free apps. If service fees on free apps are passed through to consumers, some consumers who obtained these apps for free in the actual world could pay more overall for apps in the but-for world, even if some of the apps that they paid for in the actual world would be priced lower in the but-for world. The increase in the price of free apps could be larger than the decrease in the price of paid apps. Determining whether that would be so would require calculating the price of all free and paid apps for each individual consumer in the but-for world.

38. Consumers who purchase from developers that set low price points likely would not have been impacted by Google's alleged conduct because those developers likely would not have lower cost options to Google Play in the but-for world and therefore will not set lower prices to consumers.

39. Consumers who rely on relatively costly forms of payment, such as Direct Carrier Billing[7] ("DCB") and gift cards, also likely would have been worse off in the but-for world if those forms of payment would not have been as available or available on the same terms as Google [redacted]

40. Finally, some consumers likely would have been worse off in a but-for world with less robust security warnings if security problems increased or if consumers paid more to avoid security problems.

III. ASSUMPTIONS USED TO CONSIDER DEVELOPER PLAINTIFFS' AND CONSUMER PLAINTIFFS' CLAIMS OF COMMON IMPACT

41. Developer Plaintiffs' expert, Dr. Sibley, and Consumer Plaintiffs' expert, Dr. Singer claim that in certain relevant antitrust markets, Google Play had market power and engaged in certain conduct that foreclosed opportunities to Google Play's competitors, and/or other payment processors. Both experts contend that in the but-for world Google would not have engaged in the alleged conduct and, as a result, there would be increased competition from

---

[7] DCB allows consumers to make Google Play payments as part of their mobile phone bill. For example, one of the named Consumer Plaintiffs, [redacted], paid for Google Play transactions through his [redacted]. See [redacted]

the entry and expansion of app stores and/or reliance on alternative payment processors that would lead to lower service fee rates for all putative developer class members.[8]

42. For purposes of this report, I assume Dr. Sibley's and Dr. Singer's alleged relevant antitrust markets are properly defined. I have not been asked to analyze those experts' opinions related to relevant antitrust markets and have no economic opinion as to whether their opinions on market definition are consistent with economics or the facts of this case.

43. I further assume, for purposes of my analysis, Dr. Sibley's and Dr. Singer's claim that Google Play had market power in their claimed relevant markets. However, I do not assume or concede that app distribution in the actual world was not competitive, that developers did not have alternatives to Google Play, or that developers did not, or could not, negotiate competitive service fee rates. Indeed, as described below, both Dr. Sibley and Dr. Williams – for Developer Plaintiffs – and Dr. Singer – for Consumer Plaintiffs – agree that certain putative developer class members did negotiate competitive service fee rates.

44. For purposes of this report, I assume Developer Plaintiffs' and Consumer Plaintiffs' claim that in the but-for world, Google Play would face more competition. Notably, Plaintiffs' experts do not specify what form this increased competition would take. Different sources and forms of enhanced competition could have different kinds of effects on Google's approach to the fees and services offered to different developers. Neither Developer Plaintiffs' nor Consumer Plaintiffs' experts have provided any basis to conclude that any increased competition in the but-for world would have been sufficiently strong to result in Google reducing service fees, increasing Play Points, or offering additional tools or features, to any developer or consumer, let alone with respect to all transactions on Google Play. Nevertheless, for purposes of this report, I assume (but do not concede) that increased competition would have resulted in some change by Google in order to analyze whether Plaintiffs' experts have set forth common proof that such a change would have made all or nearly all members of the proposed classes better off.

45. The questions I analyze are whether Developer Plaintiffs have established a method to show that all or nearly all putative developer class members would have been better off in a world where Google did not engage in the challenged conduct and whether Consumer Plaintiffs have established a method to show that all or nearly all proposed consumer class members would have been better off in a world where Google did not engage in the challenged conduct. For the reasons described below, I find that neither Developer Plaintiffs nor Consumer Plaintiffs have established any methodology to show classwide impact for their respective proposed classes.

8 Developer Plaintiffs' expert, Dr. Williams, also found that the alleged conduct caused putative developer class members to pay an elevated service fee rates assuming Dr. Sibley's market definitions. Williams Report at ¶¶ 8, 11.

IV. THE PROPOSED CLASSES AND PRODUCTS INVOLVED ARE HIGHLY DIFFERENTIATED

A. Apps and IAPs

1. Apps are numerous and highly differentiated

46. Software applications ("apps") are available on mobile phones, tablets, PCs, game consoles, and other hardware devices ("devices") that use different operating systems such as Windows OS, Apple iOS, and Android OS.[9]

47. Apps are highly differentiated. Many apps are available on multiple device types and multiple operating systems. For example, the Netflix app is available on devices such as smartphones, tablets, smart TVs, game consoles, set-top boxes, and Blu-ray players, among others.[10] The Netflix app is available for Apple iOS, Android OS, and Windows OS.[11]

48. Apps are created by developers and can be shared with consumers in many ways, including through app stores and direct downloads from the Internet.[12] There are several Android app stores, including Google Play, the Samsung Galaxy Store, the Amazon Appstore, and Aptoide.

49. Apps can be broadly divided into categories. Among apps listed in Google Play, 87% are listed by their developers in non-game categories.[13] Education is the largest category, followed by business, tools, music and audio, and entertainment.[14] In Google Play, there are also different categories of game apps.[15]

50. There are numerous apps within each category, and apps in each category may or may not be targeted at the same users. For example, as of May 5, 2021, there were 183,074 different shopping apps in Google Play.[16] Within the category of shopping apps, the FootLocker app

---

9 See, for example, "A Beginner's Guide to Mobile Apps," Lifewire, https://www.lifewire.com/what-are-apps-1616114 (explaining that apps are available on various device types, including desktop, mobile, and web); "Five Common Operating Systems," Small Business Chron, https://smallbusiness.chron.com/five-common-operating-systems-28217.html; "Apps for Everyone," Windows, https://www.microsoft.com/en-ca/windows/windows-10-apps (showing that Windows 10 apps are available for various Windows devices including mobile, desktop, and Xbox).

10 See https://devices.netflix.com/.

11 See https://help.netflix.com/en/node/101653.

12 See https://developer.android.com/studio/publish.

13 See Exhibit 1. This figure is based on Android apps in Google Play on May 5, 2021. The figure omits Android apps not registered in Google Play and the breakdown could be different for Apple iOS apps or Microsoft Windows apps.

14 See Exhibit 2.

15 See Exhibit 3.

16 See Exhibit 2.

and the Nike app may be targeted at the same group of consumers, but the Home Depot app may be directed to a different group of consumers.[17]

51. Apps differ along other dimensions, including the characteristics of the consumer's device, e.g., the type of processor or the amount of memory.[18] Some apps depend on having many interacting users and are differentiated by the size of their user community.[19] Many apps are available on multiple platforms or devices while others are not.[20]

52. Apps are offered by a variety of developers. For example, federal and local governments, banks, universities, and service providers such as taxi companies and ride-sharing companies all offer apps.[21]

53. Developers that choose to monetize their apps have several options for doing so. Monetization strategies include, by way of example, requiring a payment to download the app (a "paid app"), selling a subscription that recurs over time to access content in the app (a "subscription app"), selling other non-subscription digital content or features ("in-app products" or "IAPs") within the app (an "IAP app"), and earning revenue through displaying advertisements in the app.[22]

---

17 https://play.google.com/store/apps/details?id=com.footlocker.approved; https://play.google.com/store/apps/details?id=com.nike.omega; https://play.google.com/store/apps/details?id=com.thehomedepot.

18 For example, Epic Games describes the minimum device specifications for its Fortnite app as "devices running 64-bit Android on an ARM64 processor, Android OS 8.0 or higher, minimum 4GB of RAM, and GPU: Adreno 530 or higher, Mali-G71 MP20, Mali-G72 MP12, or higher." https://www.epicgames.com/fortnite/en-US/faq.

19 Similarly, certain mobile gaming apps are referred to as Massively Multiplayer Online, in which many users – sometimes thousands – play against or with one another at the same time (e.g., Arcane Legends, published by Spacetime Studios). See "What Is an MMO?" Lifewire, https://www.lifewire.com/what-is-an-mmo-4687003, accessed February 24, 2022; Arcane Legends MMO-Action RPG, Google Play, https://play.google.com/store/apps/details?id=sts.al, accessed February 24, 2022.

20 Epic's Fortnite Battle Royale, for example, is available on PCs at Fortnite.com, on game consoles through PlayStation Store, Xbox Marketplace and Nintendo eShop, and on Android mobile devices through the Samsung Galaxy Store and epicgames.com. "FAQ," Fortnite, https://www.epicgames.com/fortnite/en-US/faq, accessed January 5, 2022. Other games are available only on one platform. See, for example, a list of Nintendo Switch-exclusive games: Kamen, Matt et. al, "Best Switch exclusives to make sure you have in your library," Gamesradar, accessed January 20, 2022, https://www.gamesradar.com/best-switch-exclusives/.

21 See, for example, U.S. federal government apps, https://www.usa.gov/mobile-apps; County of San Diego apps, https://www.sandiegocounty.gov/content/sdc/dmpr/gfx/appcenter/index.html; Bank of America, https://promotions.bankofamerica.com/digitalbanking/mobilebanking; University of Connecticut, https://mobile.uconn.edu/applications/; San Francisco taxi mobile apps, https://www.sfmta.com/taxi-mobile-apps; https://play.google.com/store/apps/details?id=me.lyft.android.

22 "Monetize with ease," Google Play Console, https://play.google.com/console/about/monetize/, accessed March 17, 2022.

54. Free apps – that is, apps that are free to download and include no subscriptions or IAPs – account for about [redacted] of the apps in Google Play as of May 2021.[23] Some developers may be able to reach more users when they distribute apps as free to download, but then monetize such freely distributed apps by converting some of the users into paid users through the sale of IAPs or subscriptions.[24] Other free apps monetize through advertising or through the sale of physical products.

55. Each developer may choose whether to offer its app as a free download with no advertisements (e.g., banking apps, government apps), as a free download with paid advertising (e.g., Facebook), as a paid download (e.g., Minecraft), or as a subscription-based app (e.g., the New York Times app) or as an app with IAPs (e.g., Candy Crush).[25]

56. Currently, most of the money consumers spend (on digital content) in mobile apps ("consumer spend") is in the form of subscriptions and IAPs.[26] Subscriptions and IAPs provide consumers with recurring access to content, extra digital content and features (e.g., premium content, digital goods, digital currency) or the ability to proceed faster through a game.[27]

57. During the class period, August 2016 – December 2021, out of all apps with sales to U.S. consumers on Google Play, [redacted] offer subscriptions or IAPs. Those apps account for [redacted] of all U.S. consumers' spend on Google Play.[28] Similarly among the putative developer class,[29] subscriptions and IAPs account for [redacted] of consumer spend on Google Play during the class

---

23 Exhibit 1.

24 [redacted]

25 See BARD Mobile app, https://play.google.com/store/apps/details?id=gov.loc.nls.dtb, accessed March 25, 2022; Bank of America app, https://play.google.com/store/apps/details?id=com.infonow.bofa, accessed March 25, 2022; Facebook app, https://play.google.com/store/apps/details?id=com.facebook.katana, accessed March 21, 2022; Minecraft app, https://play.google.com/store/apps/details?id=com.mojang.minecraftpe, accessed March 9, 2022; New York Times app, https://play.google.com/store/apps/details?id=com.nytimes.android, accessed March 21, 2022; Candy Crush app, https://play.google.com/store/apps/details?id=com.king.candycrushsaga, accessed March 9, 2022.

26 Exhibit 4.

27 See https://developer.amazon.com/docs/in-app-purchasing/iap-overview.html#what-is-in-app-purchasing-iap, accessed March 22, 2022.

28 Exhibit 4.

29 The Developer Plaintiffs define their proposed class as "U.S. developers," without describing how such developers can be identified. The problems associated with the class definition are discussed in Appendix C. For the purposes of this report, "putative developer class members" are identified based on Google's App-level spend data for Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689.

period.[30] Examples of subscriptions include access to digital content in apps such as Disney+ or the *New York Times*. Examples of IAPs include digital currency like "VBucks" in the Fortnite game app or "Robux" in the Roblox game app,[31] and points that allow the player to increase the chance of success of winning the game such as "COD Points" in the Call of Duty mobile game app.[32]

58. Developers set prices of apps, subscriptions and IAPs. On Google Play, prices set by developers selling to U.S. consumers varied from $0.01 to $1,341.90 between August 2016 and July 2021.[33] Developers (and, at times, Google) offer consumers price discounts, which range from less than 1% to 100%[34] and tend to be of relatively short frequency. During the class period, 60% of promotions applied to prices for one month or less.[35]

---

30 Exhibit 4.

31 Fortnite's VBucks can be purchased on multiple platforms and can be purchased on one platform, such as Android, and spent on another platform, such as Microsoft's Xbox – a gaming console platform. See https://www.epicgames.com/fortnite/en-US/vbuckscard, accessed March 21, 2022. The Roblox app is a game creation system, or game platform, as well as a social platform. Users purchase Robux and spend Robux to enhance their avatar identities or in-game play. Roblox developers earn Robux through their games, where the exchange rate is 100 Robux for 35 cents. See Roblox Corp. Form 10-Q, September 30, 2021, at pp. 10, 12-13, https://d18rn0p25nwr6d.cloudfront.net/CIK-0001315098/ad69fbb0-a7b7-465b-942f-53206ff42303.pdf (see sections on "Description of Business," "Roblox Platform," and "Principal Agent Considerations"); "Developer Economics," Roblox, https://developer.roblox.com/en-us/articles/developer-economics (explaining to Roblox developers that "Roblox currently uses an exchange rate of ~.0035 USD per Robux earned to calculate the amount of real currency you receive.").

32 Call of Duty is a multiplayer "shooter game," categorized as an "Action" game app in Google Play. IAPs can be used to obtain "COD Points," the game's digital currency, and for "battle passes" that reward the player as they move up through more difficult levels of the game. COD Points are exchanged for weapons and other game features. IAP prices in Call of Duty range from $0.99 to $99.99. See https://play.google.com/store/apps/details?id=com.activision.callofduty.shooter, accessed March 21, 2022 (showing Call of Duty's IAP price range on mobile and "Action" category on Google Play); https://activision.helpshift.com/a/cod-mobile/?p=all&s=cod-points-credits-and-battle-pass&f=what-is-battle-pass&l=en; https://activision.helpshift.com/a/cod-mobile/?p=all&s=cod-points-credits-and-battle-pass&f=what-are-cod-points&l=en.

33 See this report's production, which shows prices in Google's U.S. consumer transaction data; Google Transactions Data, GOOG-PLAY-007203251. See also https://support.google.com/googleplay/android-developer/answer/10532353, accessed January 5, 2022 (stating a price range of $0.99 to $400.00 for the United States); https://support.google.com/googleplay/android-developer/answer/10532353, accessed March 14, 2022 (stating a price range of $0.05 to $400.00 for the United States).

34 Exhibit 5.

35 Exhibit 6. Dr. Singer finds that promotions are small. See, e.g., Singer Table 3 (calculating an App Product Price of $3.99 and App Product Price Net of Promotions of $3.97) and Singer Report Appendix 4.

2. Google Play

59. Google Play is a two-sided platform on which developers can offer Android OS apps to consumers, and consumers can find apps offered by those developers that choose to make their apps available on Google Play.[36] Google Play was introduced in 2008 as Android Market and rebranded as Google Play in 2012.[37] As a two-sided transactional platform, Google Play provides benefits to, and facilitates interactions between, consumers and developers.[38]

60. Google Play offers services to developers regardless of whether the developer uses other app stores or platforms, and regardless of whether the developer has apps that generate consumer spend or the amount of consumer spend.[39] For instance, Google Play's discovery services, which help developers find an audience for their apps, are available to all developers, regardless of whether and how they choose to monetize their apps.[40] Developers of free apps and monetized apps also can rely on various other Google tools for developing, testing, marketing, and updating apps.[41]

---

[36] See https://play.google.com/about/howplayworks/.

[37] See https://www.androidauthority.com/android-market-google-play-history-754989/.

[38] See https://play.google.com/about/howplayworks/.

[39] Requirements to become a Google Play Developer are modest; developers are required to pay a one-time $25 fee, obtain a Google email account, and adhere to the Developer Distribution Agreement ("DDA"). https://play.google.com/about/developer-distribution-agreement.html.

[40] In general, developers obtain discovery through app stores, such as Google Play, or by purchasing advertising. Google Play services are especially important to developers that do not have a recognized reputation or brand and that do not have the resources necessary to invest in advertising and marketing themselves to promote their apps. There are several "discovery" features in Google Play including "Top Charts," "Recommended for You," "New Apps We Love," "Recently Updated," "Popular Apps and Games," "Editors' Choice," and "Trending." On a page for an individual app, Google Play highlights "similar" or related apps, as well as other apps published by the same developer. Consumers can pre-register for some apps prior to their launch and receive notification when it is available. (Android, "Pre-registration," available at https://developer.android.com/distribute/best-practices/launch/pre-registration, accessed January 17, 2021. Google Play Instant and the Try Now feature allows consumers to try an app without having to install or pay for it. (Android, "Google Play Store," available at https://developer.android.com/distribute/google-play, accessed January 17, 2021.

[41] For example, tools developers use prior to the release of the app to test the app by a small group of trusted users or by a larger group, to reduce the size of the app to save storage space on a consumer device and reduce latency that may lead to lower spend on an app, and to access and incorporate app bundles that contain the elements an app needs to install correctly on mobile devices. See https://play.google.com/console/about/closed-testing/; https://play.google.com/console/about/internal-testing/; https://play.google.com/console/about/opentesting/; https://play.google.com/console/about/app-bundle-explorer/; https://play.google.com/console/about/internalappsharing/. Google offers many other publishing tools and services to developers through Google Play. One of the services provided by Google Play is processing payments if a developer offers a paid app or IAPs in Google Play. See Appendix D for a description of other tools and features provided by Google Play to developers.

61. Consumer Plaintiffs' and Developer Plaintiffs' experts opine that Google Play is a two-sided platform that exhibits indirect network effects.[42] Indirect network effects exist where the value of the two-sided platform to users on each side of the platform increases when the number of users on the other side of the platform grows.[43] For purposes of this report, I assume that Plaintiffs' experts are correct with respect to their opinions about Google Play as a two-sided transaction platform that exhibits indirect network effects. I note that Plaintiffs' experts do not properly account for the dynamics of two-sided platforms that they identify, including positive feedback effects that flow from one side to the other, dynamically in and over time. They do not simultaneously analyze the effect of any response by Google with respect to developers and any response by Google with respect to consumers. Instead, in attempting to analyze the effects of Google's response with respect to one side of Google Play, they hold constant any response from the other side.

62. Google charges for the services Google Play provides through a service fee. Google collects a service fee on each transaction for a paid app download, subscription purchase, or IAP purchase from an app distributed on Google Play. Google Play's service fee is assessed as a percentage of the transaction amount. Google does not collect a service fee from apps that do not offer paid downloads, subscriptions, or IAPs. Google also does not collect a service fee from subscriptions or other purchases made outside of the app, such as through the developer's website, even if the digital content made available by the purchase is consumed within an app distributed through Google Play. In Google Play's App Catalog 87% of global developers offered only free apps – that is, apps that require no payment to download and do not include any type of subscriptions or IAPs – as of May 2021.[44] There are no Google Play service fees associated with those developers' apps.

63. During the class period, Google Play service fee rates varied across developers and have changed over time.

64. For example, Google's Living Room Accelerator Program ("LRAP") was introduced in 2016 for developers of subscription video apps – such as [redacted] and [redacted].[45] The LRAP program apps have had a service fee rate of 15% since they began participating in the program. In 2021, Google introduced the Transactional Video Accelerator Program ("TVAP"), which expanded LRAP to on-demand television apps and has a service fee rate of 15%. The Books and Comics Accelerator Program ("BCAP"), which was also introduced in 2021, offers a 15% service fee rate to developers with books and comics apps.[46]

---

42 Singer Report at ¶19; Sibley Report at ¶¶32 – 33; Williams Report at ¶95.

43 Evans, David S., "The Antitrust Economics of Multi-Sided Platform Markets," *Yale Journal of Regulation*, Vol. 20, 2003, pp. 325-381 at 332; Singer Report at ¶19; Sibley Report at ¶114.

44 Exhibit 7.

45 GOOG-PLAY-001291192. Exhibit 8 provides a list of the developers associated with five of these programs. In 2020, Google proposed to expand the LRAP program to live TV app developers with its Living Room Accelerator Program++ ("LRAP++"). LRAP++ was designed for Live TV developers, such as [redacted]. The service rate for developers through this program is [redacted]. See GOOG-PLAY-000236162.

46 See GOOG-PLAY-006817773.R.

65. [redacted][47]

66. The Subscribe with Google ("SwG") program reduced the service fee rate for news app developers to 15%.[48] For example, the service fee rate for [redacted] app fell to 15% as part of the SwG program.[49]

67. The App Velocity Program ("AVP") is a program directed at certain app developers. As part of the program, Google provides services that effectively reduce participating developers' service fee rate by as much as [redacted].[50] For example, one developer Google analyzed for the program had consumer spend of [redacted] on Google Play. Google planned to provide marketing and promotions, ads credit, and cloud credits worth [redacted]. That would reduce the service fees associated with that developer's apps from [redacted] (based on a 30% service fee rate) to [redacted] – effectively reducing the service fee rate from 30% to [redacted].[51]

68. Project Hug/Games Velocity Program ("GVP") similarly provides expanded services to particular developers. These additional services, which can vary across developers, effectively reduced service fees. By 2021, Project Hug included [redacted] particular developers, accounting for about [redacted] of Google Play consumer spend worldwide.[52] [redacted] and [redacted] are among the developers that participated in Project Hug.[53] From an economic perspective, the value of the services provided to Project Hug developers results in effective service fee rates that are lower than the "nominal" rates applied to those developers' transactions.[54]

69. The "Google Play Partner Program for Games" is another program directed to certain top game app developers for access to additional growth tools and services tailored to the

---

47 See this report's production.

48 GOOG-PLAY-000604733; GOOG-PLAY-003335786.R; GOOG-PLAY-003331764 at -767 (list of LRAP (Living Room Accelerator Program) video app members as of July 2020).

49 See Figure 3.

50 GOOG-PLAY-003333689

51 GOOG-PLAY-003333689

52 GOOG-PLAY-001291192; GOOG-PLAY-006998204R at 206.R (2021 deck showing that Hug apps accounted for [redacted] of Play spend); GOOG-PLAY-000236162.

53 See Exhibit 8 for a list of developers that participated in Project Hug.

54 See GOOG-PLAY-000237766 showing that services provided to the Project Hug developers varied across developers and that the value of the services accounted for millions of dollars. For example, services described as "community development" were the primary type of services provided to [redacted], "user acquisition" services were the primary type provided to [redacted]; and "move to mobile" services were the primary type provided to [redacted]

developers' particular needs.[55] Eligible developers, which would include about 800 developer accounts, are those with game apps that have at least $5 million in consumer spend per year.[56] Services include a priority publishing queue, insights into device attributes across the Google Play device ecosystem, enhanced pre-launch tools, pre-enrollment of top eligible apps into Google Play Points, as well as other features that are important to and work well for game app developers.[57] From an economic perspective, these enhanced and expanded services provided to this set of developers would effectively reduce their service fee rates.

70. Google offered other sets of developers services and features tailored to them. Google offered "AAA developers" – developers with successful apps on PCs and consoles but new to mobile platforms – dedicated engineering and technical support for pre-launch and accelerated launches.[58] Google offered marketing and business growth services to [redacted] to [redacted] "Core developers" with more than [redacted] in monthly consumer spend and "strong revenue growth potential.[59] And Google offered "Funded strategy developers" – developers with operations funded (by venture capital or other sources) and led by "top talent" – with support early in their product life cycles.[60] These programs for particular developers reflect the diversity of app developers and the varying attractiveness they have to app stores.

71. Google also implemented service fee rate changes at various points in the class period. In 2018, for instance, Google reduced the service fee rate for subscriptions to 15% beyond the first year of a subscription. Effective January 1, 2022, the 15% rate applies to all subscriptions.[61] And in 2021, Google announced that starting on July 1, 2021, for those

---

55 The program had been considered as early as 2019 and is currently scheduled to launch in 2022. Deposition of Kobi Glick, December 15 and 16, 2021 ("Glick Dep.") at pp. 310-311, 329-330, 338; GOOG-PLAY-000560397 ("There are two key catalysts for the recent developer Challenges. First, while 30% rev share is the industry standard across digital stores (Steam, PSN, Xbox Live, iOS etc.), new stores start offering lower rev share structures.")

56 Glick Dep. at p. 317.

57 https://play.google.com/console/about/partnerprogram/. See also Glick Dep. at pp. 311-315.

58 GOOG-PLAY-000271389 ("Develop a global AAA strategy working group to surface, track, and share best practices across regions and xPAs + xFNs, Co-create playbook to accelerate PC-to-mobile launches, Dedicated NA Play Eng prelaunch / testing support; Need enhanced device coverage insights for new PC entrants, FTE in MTV to support prelaunch activations (e.g., exclusive access testing, closed alpha, closed beta, open / pre-reg, etc.").

59 GOOG-PLAY-000271389 ("Deliver YY scalable consultations for portfolio developers to help them grow beyond their key markets and demographics; Build a new developer onboarding process on Play with Dev Marketing, Product Specialist teams").

60 GOOG-PLAY-000271389.

61 See e.g., "[Update: Now in effect] Google raises subscription revenue for providers from 70% to 85%, but only for users retained after 12 months," Android Police, January 2, 2018, https://www.androidpolice.com/2018/01/02/google-raises-subscription-revenue-providers-70-85-users-retained-12-months/, accessed November 10, 2021; "Google Play is lowering its developer fees for app subscriptions," XDA Developers, October 21, 2021, https://www.xda-developers.com/google-play-is-lowering-its-developer-fees-for-app-subscriptions/, accessed March 25, 2022.

developers that enroll in Google's program,[62] "the service fee Google Play receives when a developer sells digital goods or services" would change to "15% for the first $1 million (USD) of revenue every developer earns each year."[63] Over 98% of putative developer class members earned $1 million or less over the class period; therefore, this change lowered the service fee rate to 15% for nearly all developers including all of the named Developer Plaintiffs, once they enrolled in the program.[64]

B. The Proposed Classes

72. In separate matters, two proposed classes – a proposed class of U.S. app developers ("Developer Plaintiffs") and a proposed class of U.S. app consumers ("Consumer Plaintiffs") – allege that certain Google policies and agreements violate federal and state antitrust laws. Both sets of Plaintiffs claim that (1) certain agreements between Google and mobile device original equipment manufacturers ("OEMs"), mobile network operators, and developers, (2) Android's security warnings to consumers and other alleged technical restrictions, and (3) Google's requirement that developers use Google Play Billing for IAPs and subscriptions in apps distributed through Google Play, enabled Google to obtain and maintain monopoly power within and cause competitive harm to alleged relevant markets.[65] Consumer Plaintiffs claim that the alleged conduct caused competitive harm during the class period from August 16, 2016 to the present.[66] Developer Plaintiffs' expert states that the relevant class period for those Plaintiffs is August 17, 2016 to the present.[67]

73. For purposes of my analysis, I assess whether it can be proved, with the same evidence for all members of each proposed class, that all or nearly all proposed class members were impacted by Google's allegedly anticompetitive conduct. I also assess whether individual damages can be determined through common proof.

1. The Proposed Developer Class Definition Includes Over 49,000 Diverse Developers

74. The proposed developer class consists of at least 49,000 U.S. developers. Developer Plaintiffs describe their proposed class as:

---

62 https://support.google.com/googleplay/android-developer/answer/10632485, accessed January 5, 2022.

63 "Boosting developer success on Google Play," Android Developers Blog, March 16, 2021, https://android-developers.googleblog.com/2021/03/boosting-dev-success.html.

64 Exhibit 9; see this report's production for the 2021 consumer spend of the named Developer Plaintiffs.

65 Developer Complaint at ¶¶8, 122; Consolidated Second Amended Class Action Complaint, December 3, 2021 ("Consumer Complaint") at ¶¶5-15.

66 Consumer Complaint at ¶213.

67 Sibley Report at fn. 1.

"All U.S. persons or entities that paid Google a 'service fee' of greater than 15% on any paid Android OS app or paid in-app content (including subscriptions) sold in or via the Google Play store, in or via any U.S. or foreign Google Play storefront."[68]

75. Developer Plaintiffs' experts use a different class definition in their analysis: "a class limited to developers that paid Google a service fee at the 30% level on at least one transaction."[69]

76. Developer Plaintiffs have not defined what constitutes a "U.S." developer in their proposed class definition, and their experts fail to investigate or provide an accurate identification of "U.S." developers. For purposes of this report, unless otherwise indicated, I consider developers that are identified in Google's App-level spend data[70] with the country code of "U.S." to be "U.S." developers. The App-level spend data identifies 49,387 developers with country code "U.S." that had at least one consumer transaction for paid downloads, subscriptions, or IAP sales between August 2016 and December 2021.[71] Note that the total number of "U.S." developers (however defined) is far greater, as many developers do not offer paid downloads, IAPs, or subscriptions. I adopt the language in this report "putative developer class members" to refer to the developers identified by the country code of "U.S." in Google's App-level data. My use of this term is not intended as an affirmative statement as to how one should circumscribe Developer Plaintiffs' class definition. As I discuss in Appendix C, the identification of a "putative developer class member" requires individualized analysis.[72]

---

68 Developer Complaint at ¶244.

69 Williams Report at ¶14, Sibley Report at ¶7.

70 The data in GOOG-PLAY-005535885 - GOOG-PLAY-005535886 and GOOG-PLAY-010801688 – GOOG-PLAY-010801689 are referred to as "App-level spend data" throughout this report. These data include monthly information regarding apps that have sales in Google Play, including app revenue, monetization type (e.g., paid download, subscription, or in-app purchase), app category, quantity, service fees, device characteristics, and form of payments.

71 Note that this counts sales to U.S. consumers as well as to consumers in the rest of the world. See Exhibit 10. Dr. Williams assumed that a putative developer class member could be identified based on the data's developer country field; his data processing contains a similar number of developers in the proposed class and their consumer spend during the class period. As described above and in Appendix C, there is no basis for this assumption and the assumption leads to some putative class members that are not U.S. developers by any reasonable definition. See Exhibit 10.

72 Using Google data is likely not an appropriate way to identify "U.S." developers. For example, Century Games appears as a "U.S." developer in the App-level spend data but is headquartered in China and is listed in the Play Store with a non-U.S. address. See GOOG-PLAY-005535885 (the "U.S." app-level spend data contains the developer Century Games, including revenues for its app Dragonscapes, whose app package name is com.dragonscapes.global); "Century Games," LinkedIn, https://www.linkedin.com/company/century-game; "Dragonscapes Adventure," Google Play Store, https://play.google.com/store/apps/details?id=com.dragonscapes.global (Century Games app listing its address to be in Singapore). In another example, Electronic Arts is identified in the App-level spend data with the country code of Netherlands. Yet Electronic Arts has been incorporated in the U.S. since 1982 – first in California and later in Delaware – and its principal executive offices are in California. Electronic Arts Inc. Form 10-K, 2020 at p. 6. In the App-level spend data, subsidiary companies of the same parent

77. Members of the putative developer class vary widely by the type and number of apps sold, monetization strategies employed, amount of revenues, competitive conditions, availability and use of alternative app distribution channels, options for processing consumer payments, service fees associated with their apps, and other factors. As a group, there were 132,593 paid apps, subscription apps, or apps with IAPs offered by putative developer class members for which there was at least one consumer transaction during the class period.[73] In addition, putative developer class members offered 76,513 free apps (that is, apps that are free to download and contain no IAPs and no subscriptions).[74] In total, putative developer class members offered 209,106 apps in Google Play.[75] In any year during the class period, approximately 20,000 of these developers offered approximately 60,000 paid download apps, subscription apps, or apps with IAPs.[76] As Figure 1 below shows, 70% of putative developer class members offered a single paid, subscription app, or IAP app while a small percentage – less than 2% – offered 21 or more.

---

are identified as separate developers. In some cases, a company and its subsidiary may have different country locations. In addition, there is ambiguity regarding whether developer refers to the creator of the app or a publisher of the app which for some apps are not the same entity. Google data only includes publishers, does not identify whether there is a separate creator, and does not have information on the terms of the relationship between publishers and creators where such relationship exists. See Appendix C for details.

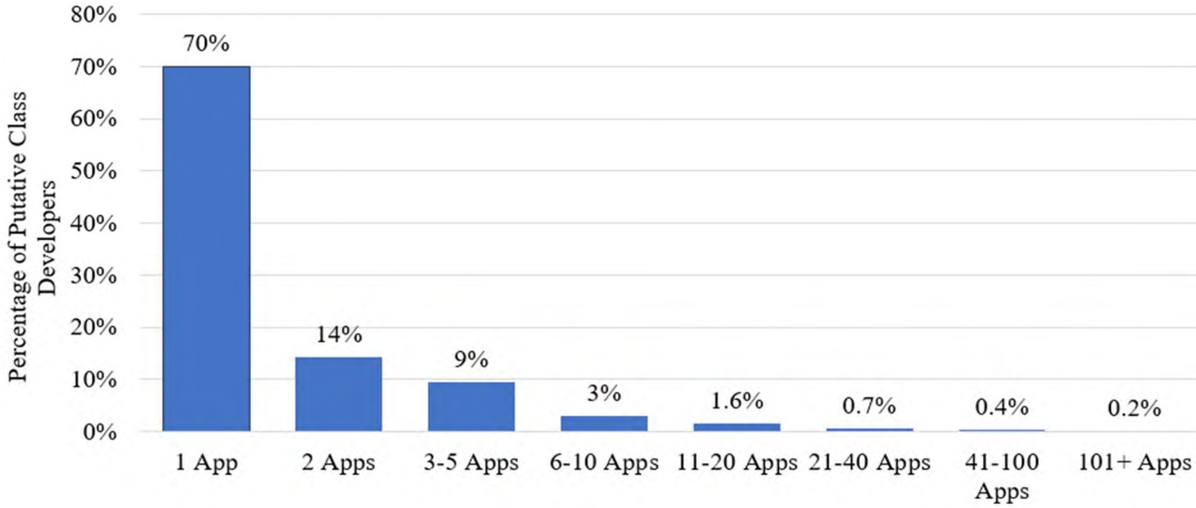
73 Exhibit 11. There were over 17,000 additional apps that were offered by putative developer class members that were paid apps, subscription apps, or IAP apps but for which there were no consumer transactions. It is possible that some of these apps, those that were free to download but contained IAPs, were downloaded for free and used by consumers without any consumer purchasing an IAP. In addition, some of these apps contained advertising and provided benefits to the putative developer class members in the form of advertising revenues. See this report's production.

74 Exhibit 1.

75 Exhibit 1.

76 Exhibit 11.

Figure 1. Percentage of Putative Developer Class Members Offering Different Numbers of Paid Apps, Subscription Apps, or IAP Apps, August 2016 – December 2021



Source:

Exhibit 12.

78. Developers vary widely in terms of the amount of consumer spend they generate. Over the class period, [redacted] of putative developer class members generated [redacted] or less in consumer spend, and [redacted] generated [redacted] or less.[77] The proposed class of developers also includes large developers like [redacted], and [redacted] some of which generate tens or hundreds of millions of dollars in consumer spend on Google Play. In 2021, the top 10 putative developer class parents accounted for nearly half (48%) of all putative developer class members' consumer spend, and the top [redacted] developer parents accounted for [redacted] of consumer spend.[78]

---

77 Exhibit 9.

78 Exhibit 13.

Figure 2. Top [redacted] Putative Developer Class Parents Accounting for [redacted] of Consumer Spend, 2021

[redacted]

0% 10% 20% 30% 40% 50% 60% 70% 80%

Cumulative Percentage of Spend by Parent Group

Source:

Exhibit 14.

79. While all putative developer class members, by definition, generate revenue through paid downloads, subscriptions, or IAPs, they do so in different ways. Some members of the putative developer class also generate revenue by including advertisements in their apps.

80. Table 1 below shows the number of putative developer class members that generate revenue through paid downloads, subscriptions, and IAPs. The table shows that over the class period, there were 26,490 developers that offered apps with paid downloads, 19,695 developers that offered apps with IAPs, and 10,820 developers that offered apps with subscriptions. Some developers use more than one type of monetization method. For example, 3,998 developers used both IAPs/subscriptions and paid downloads.

Table 1. Number of Putative Developer Class Members by App Monetization Type, August 2016 - December 2021

Year	Number of Putative Developer Class Members with Sales of			
	In-App Purchases	Paid Downloads	Subscriptions	Both Paid Downloads and IAPs/Subscriptions
August 2016 - December 2016	6,317	14,078	1,757	1,621
2017	8,424	17,695	2,695	2,104
2018	8,510	16,284	3,557	1,915
2019	8,264	9,856	4,505	1,554
2020	8,310	8,999	5,824	1,380
2021	8,656	8,212	8,077	1,236
August 2016 - December 2021	**19,695**	**26,490**	**10,820**	**3,998**

Source:

Exhibit 11.

81. There are also 32,654 developers, or 66% of the putative developer class, that have at least one app with advertising in addition to an app that generates revenue through paid downloads, subscriptions, or IAPs.[79] Advertising revenues can be substantial for at least some developers. For example, during the period from August 2016 to December 2021, [redacted], an app from [redacted], had [redacted] in consumer spend and [redacted] in AdMob earnings.[80] The app [redacted], by [redacted] generated just [redacted] in consumer spend during the class period but had [redacted] in AdMob earnings.[81] One of the developer class representatives, Peekya, generated [redacted] in consumer spend from paid downloads through Google Play through 2021[82] and [redacted] during the period of time when the

---

79 See Exhibit 15.

80 See Exhibit 16. Google sells advertising services through Admob. Admob is one of several companies that provide advertising services to app developers and therefore the advertising revenues described above may understate the developer's total advertising revenue. Other companies that provide services related to advertising include Facebook Audience Network (https://www.facebook.com/audiencenetwork/), Unity (https://docs.unity.com/ads/UnityAdsHome.html), MoPub (owned by Twitter) (https://www.mopub.com/en), Leadbolt (https://www.businessofapps.com/ads/leadbolt/), as well as others. Among putative developer class members, 66% use advertising in at least one of their apps, which includes earning advertising revenues that are generated from AdMob. See Exhibit 15.

81 See Exhibit 16. As described above, advertising revenue is available only for AdMob advertising; since there are more advertising service providers besides AdMob, the figures above may understate total advertising revenues.

82 See this report's production.

app was monetized through advertising.[83] There are no Google Play service fees associated with the revenue developers generate through advertising.

82. Some putative developer class members generate revenue through their apps from the sale of goods or services. For example, ride-sharing companies and food delivery apps generate revenues not related to paid downloads or IAPs and therefore, there are no service fees associated with those revenues. Likewise, some putative developer class members also generate revenue from the sale of physical goods. For example, Nike is a proposed class member by virtue of its Nike Training Club app, which sells subscriptions to fitness programs offered through the app.[84] Nike also generates revenue from its sales of clothing and merchandise through its Nike[85] and NIKE SNKRS[86] apps, which do not offer IAPs or subscriptions.

83. Some developers use Google Play to distribute apps but monetize the apps outside of Google Play, bypassing Google Play Billing in part or in whole. For example, consumers can purchase access to content from some developers' websites as well as from Google Play. The *Wall Street Journal* ("WSJ") offers an "All Access Digital" subscription to its content on its website and through its app, which is available for download from Google Play (as well as from other app stores).[87] Bloomberg similarly offers subscriptions both on its website and through an app available from Google Play.[88] Subscriptions to Marvel Unlimited, a comic-book app from Marvel, can be purchased both via Marvel's website – where no service fees are collected – and through the app that is available from Google Play.[89] Other examples of

---

83 Response to Interrogatory No. 6, Peekya App Services, Inc.'s Responses and Objections to Defendants' First Set of Interrogatories to App Developer Class Plaintiffs, *In Re Google Play Store Antitrust Litigation*, November 11, 2021, Case No. 3:21-md-02981-JD and Case No. 3:20-cv-05792-JD.

84 https://play.google.com/store/apps/details?id=com.nike.ntc. See this report's production.

85 https://play.google.com/store/apps/details?id=com.nike.omega

86 https://play.google.com/store/apps/details?id=com.nike.snkrs

87 The All Access Digital subscription provides access to WSJ.com and "WSJ mobile and tablet apps." https://store.wsj.com/shop/us/us/wsjuelnsb20/, accessed February 11, 2021. Prices and price offerings on the website versus through the app may be different. For example, on February 11, 2021, the "list" price of a WSJ All Access Digital subscription was different for the two different distribution channels – the list price on the WSJ website was $38.99 per month and the list price through the app was $32.99. In addition, the promotional offers were different across the two channels. Again, on February 11, 2021, the WSJ had three offerings on its website targeting consumers who had different preferences regarding the length of a subscription. It offered a three-month subscription for $4 per month as well as a six-month subscription and a 12-month subscription for $19.50 per month. (In each offer, if the consumer wanted to continue the subscription beyond the offer period, the list price was $38.99 per month.) On the same day, the promotion offered through the app was three months for free. (If the consumer wanted to continue the subscription after the three months, the price would be $32.99 per month.)

88 The website price varied depending on whether the consumer had previous experience (e.g. some "cookie") with the Bloomberg site. If the consumer did not have such experience, the offer was $1.99 for one month.

89 https://play.google.com/store/apps/details?id=com.marvel.unlimited; https://www.marvel.com/unlimited

apps that allow for payment for digital content through the app and on websites include Tinder (a dating app),[90] Evernote (an app for note taking, organizing, and task management),[91] MyFitnessPal (a fitness and dieting app),[92] and HBO Max (a video entertainment app).[93] One of the developer class representatives, Pure Sweat Basketball, offered subscriptions only through its website from 2014 through January 2019, after which it switched to offering subscriptions only through its app.[94] Another class representative, LittleHoots, offers subscriptions through its app and also offers gift subscriptions through its website.[95]

84. Service fee rates for putative developer class members also vary across developers and over time. Service fee rates for certain apps offered by putative developer class members that participate in Google's LRAP, LRAP++, ADAP, and SwG programs are 15% [redacted].[96] In 2021, there were [redacted] members of the putative developer class participating in these programs, accounting for about [redacted] of putative developer class members' total consumer spend.[97] For example, by September 2017, the ADAP program led to a reduction in [redacted] service fee rate from 30% to 15% (which continued throughout the class period).[98]

85. Similar to consumer spend (which was shown in Figure 2), a relatively small number of developers in Google Play account for nearly all service fees. In 2021, for example:

- the top *five* putative developer class members (accounting for developers in the data with common ownership) accounted for *30%* of the service fees associated with putative developer class members;
- the top *10* putative developer class members accounted for *50%* of service fees;
- the top *30* putative developer class members accounted for over *77%* of service fees; and

---

90 https://tinder.com/en-GB/feature/subscription-tiers; https://www.help.tinder.com/hc/en-us/articles/115003356706-How-do-I-create-a-Tinder-account-

91 https://evernote.com/compare-plans

92 https://www.myfitnesspal.com/premium?source=menu_bar

93 https://www.hbomax.com/ways-to-get

94 See Response to Interrogatory No. 3, Pure Sweat Basketball, Inc.'s Responses and Objections to Defendants' First Set of Interrogatories, *In re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD, November 19, 2021, p.7-9 (showing subscription purchases in the app starting in 2019; Pure Sweat Basketball, https://web.archive.org/web/20170214231549/https://puresweatbasketball.com/training-app/ (showing that Pure Sweat Basketball offered subscription purchases on its website in 2017).

95 See "Shop," LittleHoots, https://www.littlehoots.com/shop/p/ftxisgkwjypb7d6v7zgsxrae8saj8l, accessed January 26, 2022 (showing a purchase option for a "1-year LittleHoots Subscription Gift Certificate"); Play App Catalog, GOOG-PLAY-001507601 (showing that LittleHoots' app was offering subscriptions as of May 5, 2021).

96 Google's LRAP++ program provided a service fee rate of [redacted]. See GOOG-PLAY-000236162.

97 Exhibit 17. A list of the developers associated with these programs is attached as Exhibit 8.

98 See this report's production.

- the top *105* putative developer class members accounted for *90%* of the service fees.[99]

86. Developers (and apps) also vary in terms of the value they obtain from Google Play features and benefits. Google documents show that the type and extent of benefits from Google Play features vary across apps and developers.[100] The types of benefits Google provide include transaction processing (including that some types of transactions, such as DCB and gift cards, provided some developers with incremental consumer spend); the benefits of having consumers discover and re-engage with apps, provided through promotions, features, and programs such as Google Play Points; consulting services; as well as other benefits. Google has estimated, for example, that nearly half of the value it quantified for [redacted] was generated from the availability of DCB and gift cards while for [redacted], the uplift from Google Play Points generated about 60% of the quantified value.[101] The documents provide evidence that that different developers obtain varying degree of value from different Google Play services and features.

87. Differences across developers and apps matter to the determination of common impact. The differences in characteristics show that developers have different value to app stores, including Google Play. Developers with popular apps or apps that generate substantial amounts of consumer spend are able to attract consumers to an app store and thus, those developers have relatively more bargaining power to negotiate terms with apps stores. These differences would continue to exist in a but-for world and must be considered in evaluating the effect of an increase in app store competition. The differences suggest that any increase in competition in a but-for world would have the same disparate effect as competition does in the actual world.

[99] Exhibit 18. Exhibit 19 provides data for the class period without considering the changing ownership of developers over those years: over the class period, there are 157 "U.S." developers identified in Google data that account for 90% of the service fees paid by the "U.S." developers. That is, less than 1% of developers account for 90% of the service fees. See GOOG-PLAY-005535885 and GOOG-PLAY-010801689.

[100] GOOG-PLAY-011023692 at 705, 712; GOOG-PLAY-000286913. Google has attempted to quantify certain benefits to some developers from Google Play, recognizing that such quantification is complex because of the different benefits available to developers as well as the differences across developers. While precise estimates of Google Play's value to individual developers are not available, the documents demonstrate that Google Play has many different ways to provide value to developers and that developers, because of their differences, find certain services and features have varying value to them.

[101] See also GOOG-PLAY-000286913 (showing the value of distributing and updating Facebook's app accounts for a substantial amount of quantified value because of the number of Facebook users and the frequency of the app updates).

2. The Proposed Consumer Class Definition Includes Over 90 Million Distinct Consumer IDs

88. The proposed consumer class consists of over 92 million consumer IDs in the U.S. associated with at least one purchase of a paid app, a subscription or IAP in Google Play during the class period.[102]

89. Consumer Plaintiffs describe the proposed Nationwide Class as:

> "All persons in the United States who paid for an app through the Google Play Store, or paid for in-app digital content (including subscriptions and/or ad free versions) on an app that was offered in the Google Play Store from August 16, 2016, to the present."[103]

90. Consumer Plaintiffs describe the proposed Repealer-State Class as

> "All persons in those states whose laws permit indirect purchaser standing and provide for antitrust recovery to indirect purchasers, who paid for an app through the Google Play Store, or paid for in-app digital content (including subscriptions and/or ad free versions) on an app that was offered in the Google Play Store from August 16, 2016, to the present."[104]

91. In this report, unless otherwise described, analyses and statistics related to the proposed class of consumers relate to the proposed nationwide class.

92. Members of the proposed consumer class vary by the type and number of apps for which they made purchases, types of purchases made, amount of money spent on Google Play, and reliance on free apps.

93. Over the class period, there were over 92 million U.S. consumer IDs associated with at least one purchase in Google Play of a paid app, a subscription or an IAP. The purchases involved over 404,000 unique apps.[105] Table 2 shows that through July 3, 2021, 41% of U.S.

---

102 Exhibit 20. The number of consumer IDs is based on Google's transaction-level database. Each consumer ID is identified in the data with a "hashed purchase initiator Google ID." Some consumers have more than one ID. For example, one of the class representatives, Mr. Matt Atkinson has two IDs (see this report's production). Determining which IDs are linked to which consumers requires inspection of detailed information (e.g., name, address, email address). As described below, a consumer is harmed only if the total cost of their purchases is lower in the but-for world compared to the actual world. Therefore, to demonstrate harm for a consumer, it is necessary to identify all the consumer's purchases. However, because some consumers have multiple consumer IDs and there is no simple way to link different IDs held by the same consumer, there is no way to identify the full set of apps downloaded and the total spending on apps, subscriptions, and IAPs for any consumer and no way to consider their total cost of purchases in the actual and but-for worlds to determine impact. In this report, I discuss "consumers" when, in fact, the data represent consumer IDs.

103 Consumer Complaint at ¶213.

104 Consumer Complaint at ¶213.

105 See Exhibit 20. Approximately 9 million consumer IDs were excluded from the class definition during the class period if they purchased only free trials or had only refunds. See this report's production.

consumers made purchases of or from only one app and over two-thirds of U.S. consumers made purchases of or from three apps or less. On the other side of the spectrum, over 89,000 U.S. consumers have made purchases from more than 100 apps.

94. As I describe below, whether a developer would be subject to a lower service fee and whether that developer would pass on such a lower service fee in the form of a lower price are both individual questions not subject to common proof. For any putative consumer class member that is part of the 41% of U.S. consumers that made purchases of or from only one app, if the developer of the app would not be subject to a lower service fee or would not lower the price of the app, subscription, or IAPs in the but-for world, that class member has not been impacted by Google's alleged conduct. But the only way to make that determination is to examine the particular app for that individual.

Table 2. Number of Apps from which U.S. Consumers Purchase during August 2016 – July 3, 2021

Number of Apps	Number of Consumers	Percent of Consumers	Cumulative Percent of Consumers
Only 1 App	37,690,509	41%	41%
2 Apps	15,369,289	17%	57%
3 Apps	8,932,225	10%	67%
4 Apps	5,921,644	6%	74%
5 Apps	4,240,558	5%	78%
6 Apps	3,177,880	3%	82%
7 Apps	2,468,276	3%	84%
8 Apps	1,962,423	2%	86%
9 Apps	1,594,409	2%	88%
10 Apps	1,313,045	1%	90%
>10 to 50 Apps	9,101,126	10%	99%
>50 to 100 Apps	468,283	0.5%	99.9%
More than 100 Apps	89,601	0.1%	100%
Total	**92,329,268**	**100%**	**100%**

Source:

Exhibit 21.

95. In addition to using apps with paid downloads, subscriptions, or IAPs, members of the proposed consumer class also download apps that are free, including apps that include advertising. The class representatives for Consumer Plaintiffs installed numerous free apps on their devices. For example, [redacted] installed [redacted] free apps across [redacted] devices out of a total [redacted] installed apps, and [redacted] installed [redacted] free apps out of a total of [redacted] installed apps.[106]

96. The amount of consumer spend varies substantially across U.S. consumers. Table 3 below shows that [redacted] of putative consumer class members spent less than $5 during the class

106 Exhibit 22 (The free app counts exclude pre-installed apps by carriers, OEMs, and Google).

period and about [redacted] spent less than [redacted] during the class period. Only [redacted] of the putative consumer class members spent [redacted] or more. Approximately [redacted] U.S. consumers spent more than [redacted] on Google Play during the class period.

Table 3. Distribution of Dollars Spent Across U.S. Consumers during August 2016 – July 3, 2021

Consumer Spend	Number of Consumers	Percent of Consumers	Cumulative Percent of Consumers	Cumulative Percent of Spend
Less than $5	14,342,706	16%	16%	0.1%
$5 - <$10	10,006,443	11%	26%	0.3%
$10 - <$25	12,792,164	14%	40%	0.9%
$25 - <$50	11,156,569	12%	52%	2%
$50 - <$100	11,570,016	13%	65%	4%
$100 - <$200	10,623,145	12%	76%	8%
$200 - <$300	5,157,216	6%	82%	11%
$300 - <$400	3,077,804	3%	85%	14%
$400 - <$500	2,114,207	2%	88%	16%
$500 - <$1,000	4,961,002	5%	93%	25%
$1,000 - <$2,000	2,976,406	3%	96%	36%
$2,000 - <$5,000	2,147,453	2%	98%	52%
$5,000 - <$10,000	820,572	1%	99%	67%
$10,000 - <$20,000	383,927	0.4%	99.8%	80%
$20,000 - <$50,000	165,402	0.2%	99.96%	92%
$50,000 - <$100,000	26,665	0.03%	99.99%	97%
$100,000 or More	7,571	0.01%	100%	100%
All Consumers	**92,329,268**	**100%**	**100%**	**100%**

Source:

Exhibit 23.

97. In addition, the number of Google Play transactions varied widely across consumers. Approximately 21% of putative consumer class members made a single purchase in Google Play during the class period.[107] Notably, these 21% of consumers purchased (e.g., a paid download, subscription, or IAP) from more than 150,000 different apps.[108] Again, if that single transaction would not be subject to a lower service fee or would not have a lower price in a but-for world, these consumers are not injured. Determining whether these consumers were injured requires analyzing each individual transaction because these consumers made

107 See Exhibit 24.

108 See Exhibit 24.

only a single transaction. Moreover, determining impact for these consumers implicates over 150,000 different apps.

V. ECONOMIC FRAMEWORK FOR ANALYZING COMMON PROOF OF CLASSWIDE IMPACT

A. ANTITRUST IMPACT RELATIVE TO A BUT-FOR WORLD

98. Antitrust impact refers to a proposed class member's injury-in-fact resulting from the defendant's alleged anticompetitive conduct. Antitrust impact is binary: each class member either was or was not impacted. Antitrust impact is distinct from the determination of damages because it does not purport to measure the degree of harm, but only to establish the existence of harm.

99. To show antitrust impact for a class member, Plaintiffs must establish that the class member is worse off in the "actual world" as compared to a hypothetical world in which the alleged anticompetitive conduct did not take place (i.e., in the "but-for world"). Identifying the differences between the actual world and the but-for world and evaluating how those differences affect a putative class member, are essential to determining whether that putative class member experienced antitrust impact.[109]

B. COMMON PROOF OF CLASSWIDE ANTITRUST IMPACT AND DAMAGES

100. I understand that at the class certification stage, an important question is whether plaintiffs representing a putative class can show, with common evidence that all or nearly all class members experienced antitrust impact. If individualized analysis is needed to determine whether a proposed class member was impacted, or if the proposed class includes members who were not harmed, then establishing classwide impact using common proof is not possible.[110] I also understand that at the class certification stage, an important question is whether plaintiffs representing a putative class can estimate damages with a common method that takes into account the relevant and important factors that may be different across class members.

---

[109] See e.g., Johnson, John H. and Gregory K. Leonard, "Rigorous Analysis of Class Certification Comes of Age," *Antitrust Law Journal*, Vol. 77, No. 2, 2011, pp. 569-586; *Reference Manual on Scientific Evidence,* National Academies Press, Third Edition, 2011, pp. 429-430.

[110] To determine damages, the amount of antitrust harm must be quantified. If a proposed consumer class member is found to have paid a price that included an overcharge, then a damage analysis must calculate the difference between actual price and the price that would have been paid in the but-for world. If a proposed developer class member is found to have lost profits, then a damage analysis must calculate the difference between actual profits and the profits that would have been earned in the but-for world. As described below, there are several reasons that determination of impact cannot be proved with common evidence and that certain proposed class members, in both proposed classes, were likely not impacted. For those same reasons, damages to the proposed classes cannot be proved with common evidence and attempts to calculate damages with averages or aggregated amounts would mean that some members of the proposed classes would be compensated even though they suffered no antitrust impact. Given the size of the putative classes, and the number of factors affecting the amount of damages for each class member (if any), the damages calculations in this case would be extraordinarily complex. Plaintiffs' experts have not accounted for that complexity or shown a reliable method of calculating damages for each class member through common proof.

C. Plaintiffs' Theories of Antitrust Impact Depend on Proof of Service Fee Rates and Prices in a But-For World

101. The but-for world is defined by the absence of the alleged conduct: in the but-for world, the alleged conduct does not exist. Below I explain each set of Plaintiffs' theories of antitrust impact and how those theories involve two key issues: (1) whether service fee rates would have been lower in the but-for world, and (2) whether prices of apps, subscriptions, and IAPs would be lower in the but-for world. As explained below, neither set of Plaintiffs have common proof of classwide impact.

102. **Consumers.** Consumer Plaintiffs allege that Google obtained and maintained monopoly power in certain alleged relevant markets and foreclosed competition in those markets. They claim that because of its alleged monopoly power, Google has been able to impose a supra-competitive service fee rate of 30%. Consumer Plaintiffs claim that but-for the alleged conduct, Google would be compelled to lower its service fee rate to all developers, and developers in turn would reduce prices of apps, subscriptions, and IAPs purchased through Google Play.[111] Consumer Plaintiffs' expert, Dr. Singer, claims in the alternative that Google would have responded to increased competition by increasing consumer "subsidies" through its Play Points program.[112]

103. Determining whether any individual consumer was injured as a result of higher prices caused by the challenged conduct requires a highly individualized inquiry into the price of each app, subscription, and IAP in the bundle of apps the consumer downloaded or used, including free apps. If the price of the app, subscription, or IAP would have remained the same in the but-for world, there is no antitrust impact associated with that purchase. Determining whether a price would remain the same or be lower in the but-for world first depends whether the developer of the app would have been subject to a lower service fee, and then depends on whether the developer of the app reduces prices, uses the saved fees to invest in the app (or some other app), or "pockets" the difference. Developers' decisions regarding any saved fees depend on the developer's business strategy and if the developer elects to keep the saved service fees, there is no consumer benefit from the lower service fees in the but-for world. In addition, whether the developer reduces prices depends on app-specific characteristics, including but not limited to the marginal cost of the app, subscription, or IAP, and the demand elasticity of the app. Moreover, even if the price of some apps, subscriptions, or IAPs would have fallen in the but-for world, if the free apps a consumer downloaded would have gone up in price by an amount more than any alleged overcharge on consumer's purchased apps, subscriptions, and IAPs, then there is no antitrust impact – and that could be true for many class members given how many consumers made only a very small number of purchases and spent relatively small amounts of money.

104. Determining whether a consumer would be better off in the but-for world through an expansion of Google Play Points depends, at a minimum, on whether that consumer would have participated in the Play Points program and redeemed any rewards. As described below, even though the Google Play Points program is available to all consumers, many consumers choose not to participate. Even if the program were expanded, there is no

---

111 Consumer Complaint at ¶¶4-15; Singer Report at ¶33.

112 Singer Report at ¶33.

evidence that more or all consumers would benefit. Determining which consumers would benefit would require individualized information about consumer's practices and preferences in regard to Google Play Points.

105. **Developers.** Developer Plaintiffs allege that Google improperly obtained and maintained a monopoly in certain alleged relevant markets and foreclosed competition in those markets. Developer Plaintiffs claim that through the exercise of its alleged monopoly power, Google was able to impose a supracompetitive default 30% transaction fee, and required that apps, in-app purchases, and subscriptions be priced no lower than $0.99. They claim that absent the alleged conduct, Google Play's service fee rates would be lower for all developers, at least some developers would reduce their retail prices, and developers' sales would have been higher.[113]

106. Dr. Williams, one of Developer Plaintiffs' experts, claims that in the but-for world, service fees would have been lower than in the actual world and the difference is the amount the developers were overcharged.[114]

107. Both sets of Plaintiffs assert antitrust impact and seek damages based on the *same* difference in the actual and but-for service fees; that is, the two distinct groups claim the same "overcharge" as the measure of their harm.[115] For reasons that I explain below, even if it is theoretically possible that both developer and consumer class members could each claim some non-overlapping portion of the alleged overcharge for each app, determining what portion of each service fee overcharge should be allocated to which developer and which consumer would be highly complex and would require individualized analysis of each app.

108. Dr. Williams, one of Developer Plaintiffs' experts, also discusses lost profit damages. Overcharge damages are different from lost profit damages.[116] Overcharge damages consider only the difference between service fees in the actual and but-for worlds. Lost profits damages consider the additional effects of a price reduction on a developer's sales and can include changes in the developer's costs in the but-for world. To the extent Developer

---

113 Developer Complaint at ¶8, ¶14, ¶28, ¶179; Sibley ¶36, ¶¶259-261, ¶¶270-273. Developer Plaintiffs' experts do not address the allegation that Google Play's minimum price of $0.99 was anticompetitive or the extent to which retail prices would be lower absent Google Play's minimum price and any effects from those lower prices on putative developer class members. In addition, the Developer Plaintiffs' expert finds that "over 92% of product-monetization-type combinations have zero or negative pass-through rates." See Williams Report at ¶80.

114 Williams Report ¶9.

115 Some Consumer Plaintiffs purchased apps, IAPs, and subscriptions from the putative class of developers. Dr. Williams estimates that, for the period August 2016 to December 2020, damages to the putative developer class from sales to U.S. consumers range from $1.3 billion to $1.4 billion, depending on the benchmark used (and including his estimated deduction for consumer "rewards"). Dr. Singer, the expert for Consumer Plaintiffs, estimates for that same period, damages to U.S. consumers from apps purchased from the putative developer class ranges from $0.8 billion to $1.7 billion, depending on whether IAPs are considered together or separate from paid downloads. These estimated damage amounts reflect the same alleged harm and are thus duplicative claims. See this report's production.

116 Williams Report at ¶72 and Appendix III.

Plaintiffs are limited to lost profits damages, their experts have not proposed a methodology, common or otherwise, to calculate lost profits damages.

109. A lost profits analysis is even more individualized than an overcharge analysis because developers' profits can depend on their prices and their distribution costs. Thus, calculating whether a developer lost profits depends not only on the proof that a developer would have paid a lower service fee rate in a but-for world, but also on proof of a developers' prices and distribution costs in a but-for world.

110. If developers would respond to service fee reductions by reducing prices to consumers, then developers would not suffer antitrust impact unless those price reductions generate sufficient incremental sales volumes to be profitable. The amount of incremental sales volume due to a price reduction depends on that app's price elasticity of demand, which in turn depends on the availability and relative closeness of substitutes for the app. In addition, the amount of incremental sales volume depends on whether the prices of those substitute apps also fall in response to a service fee rate reduction. Similarly, whether a developer experiences higher distribution costs in the but-for world depends on characteristics of its app. As explained below, the extent to which reductions in the service fee rate for each developer are offset by changes in revenue or changes in distribution costs is highly individualized and requires information about the particular app, the competition for that app, the cost of the app to the developer (including distribution costs, which may be higher in the but-for world if the developer uses additional app stores[117]), and other factors that are particular to the app and the developer.

111. Developer Plaintiffs' determination of antitrust impact is further complicated by the need to account for the effects on free apps in a but-for world. Free apps make up a significant share of the apps available in Google Play and are present in substantial amounts in all app categories.[118] Price elasticity of demand for paid apps, subscription apps, and apps with IAPs depends on substitutability of these apps with free apps. Moreover, a significant percentage of the putative developer class offers free apps. To the extent that Google Play (or other app stores) in the but-for world would choose to charge fees for those apps those developers could be worse off. These effects too must be considered in analyzing whether a putative developer class member that also has free apps would earn higher profits in the but-for world.

VI. NEITHER SET OF PLAINTIFFS CAN ESTABLISH CLASSWIDE ANTITRUST IMPACT WITH COMMON PROOF

A. No Common Proof of Uniform Lower Service Fees in the But-for World

112. Both sets of Plaintiffs claim that in the but-for world, Google Play's service fees rates would have been lower than in the actual world. However, Plaintiffs cannot assume that every member of the proposed classes would have been subject to a lower service fee rate.

---

117 App developers with lower service fees in the but-for world could choose to reduce app prices, to invest in apps, or to "pocket" the difference in service fees. Whether a developer reduces app prices or uses any lower service fees to invest in its apps depends on the developer's business strategy. If a developer decides to "pocket" the difference in service fees, there is no consumer benefit from the lower service fees in the but-for world. That too, is an individualized decision on the part of the developer.

118 Exhibit 2 and Exhibit 3.

More than a de minimis number of proposed class members would not have a lower service fee and thus would not be injured at all; and determining which members would obtain lower service fees requires individualized analysis.

1. No Basis for Assuming a Uniform But-For Service Fee or a Uniform Service Fee Reduction

113. The fact that Google Play's service fee rate structure and rates vary across developers and over time in the actual world belies Developer Plaintiffs' and Consumer Plaintiffs' assertion that service fee rates or service fee rate reductions would be common and uniform in the but-for world.[119] Because apps vary in terms of their importance to app stores and vice versa – which has allowed certain app developers to obtain specific deals, programs, and services – individualized analysis is necessary to determine the service fee rates that would apply to each app in a but-for world.

114. Google provides certain developers with lower service fee rates and expanded services through special programs (such as LRAP, ADAP, SwG, Project Hug, and the App Accelerator Program described above). Evidence indicates that these lower rates and expanded services reflect an effort to compete against alternatives for developers to distribute or monetize their apps.[120]

115. Even taking Plaintiffs' market definitions as given, these alternative distribution channels for Android apps include (at least) the Samsung Galaxy Store,[121] the Amazon Appstore,[122]

---

[119] Dr. Williams claims that all putative developer class members would have the same service fee rate in the but-for world. See Williams Report at ¶¶44, 50. Dr. Singer claims that there could be two but-for service fee rates – one for paid apps and another rate for subscriptions and IAPs; or there could be a single but-for rate for all developers and transactions. Dr. Singer allows a category's but-for service rate to deviate from the overall but-for rate by the same proportion in which the category's actual rate deviates from the average actual rate. See Singer Report at Table 3, Table 5, Appendix 4 and ¶280.

[120] See e.g., https://developer.apple.com/programs/video-partner/ (Apple's 2016 Video Partner Program offering certain premium video entertainment developers a service fee rate of 15%); Amazon Developer Services Agreement, last updated September 25, 2018, https://web.archive.org/web/20181026122837/https://developer.amazon.com/support/legal/da (Amazon's 2018 policy providing developers a revenue share of 80% for Movies & TV subscription IAPs). Both Consumer Plaintiffs' and Developer Plaintiffs' experts contend that these rate reductions reflect competitive conditions. See Sibley Report at ¶¶102, 205; Williams Report at ¶54.

[121] The Samsung Galaxy Store comes pre-installed on all Samsung mobile devices. In a Google survey of consumers, about [redacted]s of Samsung Galaxy owners had tried the Samsung store and [redacted] used it regularly. See GOOG-PLAY-000097630 at 630-632.

[122] https://www.amazon.com/gp/help/customer/display.html?nodeId=GP96AU3MQ58FMV8U In addition to being a developer with its popular (free) e-commerce app and offering an app store, Amazon has other businesses relevant to app distribution and the servicing of developers including its game engine – Lumberyard, cloud computing services (AWS), and the physical distribution of app gift cards and game developers' physical disks. See https://docs.aws.amazon.com/lumberyard/latest/userguide/lumberyard-intro.html.

Aptoide,[123] Opera Mobile Store,[124] and a variety of app stores available to developers that sell to consumers outside of the U.S. such as ONE store[125] and UptoDown,[126]

116. Developers may also distribute apps directly on their websites – a process sometimes called "side-loading." For example, Epic Games distributes Fortnite, a popular gaming app, on its website.[127]

117. As Google's business documents indicate, Google recognizes that developers can introduce apps first in a competing app store or could sell digital content outside of their apps, thereby avoiding paying Google Play's service fee.[128] Google's response has been to offer targeted rate reductions and expanded services. As shown in Exhibit 17, there are [redacted] that participate in the programs LRAP, LRAP++, SwG, ADAP, and Hug/GVP.[129] These developers accounted for [redacted]. The global developers that participate in these programs accounted for [redacted][130]

118. Google explicitly recognizes that its strategy of offering reduced service fees or expanded services applies only to some developers. One document, for example, states: "Do not have a

---

123 See https://en.aptoide.com/company/about-us (stating that Aptoide has 300 million users worldwide, 7 billion downloads, and 1 million apps).

124 https://publishers.apps.bemobi.com//Opera-Microsoft-FAQ.html; https://www.wikiwand.com/en/Opera_Mobile_Store.

125 See ONE store Introduction, 2020.02, at slide 4, https://dev.onestore.co.kr/devpoc/static-res/files/ONEstoreIntro_dev_en.pdf, accessed July 7, 2021. ("ONE Store is embedded into almost all Android devices in Korean market, regardless of carrier. As a result, the number of devices with ONE store installed is over 50 million.").

126 See "What is Uptodown?" Uptodown, https://www.uptodown.io/what-s-uptodown, accessed November 17, 2021 (stating that Uptodown is headquartered in Spain and has "content localization based in different locations like Indonesia, India, United States, Italy, Japan, South Korea, Germany, France, Thailand, China or Russia.").

127 https://www.epicgames.com/fortnite/en-US/mobile/android/get-started See also https://www.androidauthority.com/best-sideloaded-apps-android-1155580/ for apps that are not available on Google Play but available through "side-loading." Certain app developers can and do distribute apps through PC platforms (e.g., Steam, Epic Games Store, Windows Store) or game console platforms (e.g., Microsoft Store, Nintendo eShop, and PlayStation Store). Some large game developers use their own platforms to distribute their games for PCs. For example, Electronic Arts has Origin, (https://www.origin.com/usa/en-us/store), Ubisoft has Uplay (https://ubisoftconnect.com/en-US/), Activision Blizzard has Battle.net (https://us.shop.battle.net/en-us) , and Bethesda has the Bethesda Launcher (https://bethesda.net/en/store/home). Game developers can also distribute apps through Facebook Instant Games, or through Crazy Games – a browser-based game platform (https://about.crazygames.com/).

128 See e.g., GOOG-PLAY-000542244.

129 Exhibit 17.

130 Exhibit 17.

one-size fits all offering."[131] Google's use of targeted service fee rates reductions and expanded services makes economic sense as a means to compete for developers that are particularly valuable to Google.

119. The fact that other app stores provide lower service fee rates and/or more benefits to certain targeted developers provides additional evidence that Google's competitive strategy would not change in a but-for world. The Amazon Appstore, for instance, offered "key" game developers – that is, developers of 20 of the top 100 games in Japan – lower service rates and other incentives.[132]

120. [redacted][133] [redacted][134]

121. App stores' targeted use of incentives in the actual world reflects that different apps have different value to app stores. Currently, in the actual world, developers that have particularly attractive apps to app stores can receive special incentives. There is no reason to assume that in the but-for world, these differences across developers would not continue to exist. Just as service fee rates in the actual world are not uniform, the same is likely to continue to be the case in the but-for world: service fee rate reductions are not likely to be uniform.

122. The likelihood that not all apps or developers would experience service fee rate reductions in the but-for world is further borne out by the fact that rate reductions have not been uniform across apps or developers in the actual world. Google Play has reduced its

---

131 See e.g., GOOG-PLAY-000542244 at 244. See also "Evolving our business model to address developer needs," Android Developers Blog, October 21, 2021, https://android-developers.googleblog.com/2021/10/evolving-business-model.html ("The creativity and innovation from developers around the world spurred amazing new app experiences we could have never imagined when we first introduced Android. As the ecosystem evolved, a wider range of business models emerged to support these different types of apps. We've made important changes along the way, including moving beyond a "one size fits all" service fee model to ensure all types of businesses can be successful. Instead of a single service fee, we now have multiple programs designed to support and encourage our diverse app ecosystem.")

132 GOOG-PLAY-000568027 at 028.

133 See EPIC_GOOGLE_01413875 at 877 ([redacted]).

134 See Samsung Galaxy terms and conditions 2014, https://seller.samsungapps.com/notice/getNoticeDetail.as?csNoticeID=0000002677; Samsung Galaxy terms and conditions, 2019, https://seller.samsungapps.com/notice/getNoticeDetail.as?csNoticeID=000004278; GOOG-PLAY-000561051.R at 019.R; [redacted]

service fee rates for particular types of apps, particular developers, and particular kinds of developers.

123. As described above, the LRAP program introduced in 2016 reduced the service fee rate for subscription video apps, such as [redacted] from 30% to 15%.[135]

124. Likewise, the ADAP program reduced service fee rates for audio and music app developers, but not for travel apps, game apps, or any other app category.[136]

125. In 2018, Google reduced the service fee rate for subscription IAPs to 15% after a consumer subscribes for one year, but this reduction did not apply to pay-to-download apps or to non-subscription IAPs.[137] Similarly, the SwG program reduced the service fee rate to 15% for all news subscriptions. For example, as shown below in Figure 3, the effective service fee rate for [redacted] app fell from 30% to about [redacted]% after December 2017 due to the 2018 service fee rate reduction, and then fell to [redacted]% due to the SwG program.

135 See GOOG-PLAY-000604733 at 738; GOOG-PLAY-001291192 at -208. Exhibit 8 provides a list of the developers associated with five of these programs.

136 See GOOG-PLAY-000604733 at 738. Exhibit 8 provides a list of the developers associated with five of these programs.

137 "Google Play Lowers App Subscription Fee to 15 Percent, Matches Apple's Offering," Gadgets 360, October 20, 2017, https://gadgets.ndtv.com/apps/news/google-play-app-subscription-fee-30-percent-to-15-1764923, accessed November 8, 2021. Google's subscription fee rate reduction followed Apple's 2016 rate reduction on subscriptions. See Fingas, Roger, "Apple announces it will offer App Store subscriptions to all apps, take smaller 15% cut," Apple Insider, June 8, 2016, https://appleinsider.com/articles/16/06/08/apple-announces-it-will-offer-app-store-subscriptions-take-smaller-15-cut, accessed November 8, 2021.

[redacted]

126. Similarly, the Apps Velocity Program and Project Hug program offered bespoke services to certain targeted developers, but not all developers. By 2021, Project Hug had signed [redacted] targeted developers, accounting for about [redacted] of Google Play consumer spend worldwide.[138]

127. Figure 4 shows the distribution of service fee rates for the 7,291 putative developer class members that had paid download, subscription, or IAP sales in Google Play and an average annual service fee rate greater than zero percent in every year from 2017 through 2021. Figure 4 shows that when Google reduced service fee rates, it did so for only certain developers.[139]

---

138 GOOG-PLAY-001291192; GOOG-PLAY-006998204R at 206.R (2021 deck showing that Hug apps accounted for [redacted] of Play spend); GOOG-PLAY-000236162.

139 See Exhibit 26 showing the percentages for all putative class developers with an average annual service fee rate greater than zero percent in any year between 2017 and 2021 in which they had sales in Google Play.

[redacted]

128. When other app stores reduced service fee rates, those reductions were generally limited to a targeted set of developers or apps as well. Exhibit 28 provides a summary of app stores' rate reductions and shows that in nearly all instances, when an app store has reduced service fee rates, it did so for a certain set of developers or apps, not all developers and apps.

129. There are two instances in which an app store reduced service fee rates for all apps; however, both of those cases contradict Plaintiffs' claims about the but-for world. In July 2018, ONE store, an Android app store in South Korea, reduced its 30% service fee rate to 20% for all developers and to 5% for any developer that used the developer's own payment processing system.[140] Although Dr. Singer claims that ONE store was able to "overcome the prohibitive restrictions on competition imposed by Google," and obtain 14.9% of payment volume, ONE store's rate reduction did not lead Google Play or other app stores to reduce

[140] ONE store Introduction, ONE store Corp. January 2020 at p. 9, https://dev.onestore.co.kr/devpoc/static-res/files/ONEstoreIntro_dev_en.pdf, accessed November 8, 2021. ONE store entered the app distribution business in Korea in June 2016, when three app stores, operated by three telecom companies in Korea – SK Telecom, KT and LG – integrated their stores and rebranded as ONE store. In 2020, ONE store described itself as an "Android-based distribution platform which provides mobile apps, games, eBook content, and physical goods." Since 2016, it has been pre-installed on almost all Android phones in Korea. ONE store Introduction, ONE store Corp. January 2020 at pp. 3-5, 9, https://dev.onestore.co.kr/devpoc/static-res/files/ONEstoreIntro_dev_en.pdf, accessed November 8, 2021.

service fees across the board (even in South Korea).[141] This undermines Plaintiffs' experts' assumption that more successful alternative app stores would result in across-the-board reductions in service fees. Further, in 2018, the Epic Game Store entered app distribution for PC Games with a service fee rate of 12%. The Epic Game Store's service fee rate was and continues to be lower than the rates of many other app stores.[142] The entry of the Epic Game Store did not lead to reduced service fee rates for all developers, or all game developers, or even all developers of PC games.[143] In the limited instances where an app store has offered lower rates to all developers, other app stores, including Google Play, did not respond in kind to the lower rates.[144]

2. Not All Developers Would Pay Less If Developers Could Separately Contract with Payment Processors

130. Plaintiffs' claim that all developers would pay less because, in the but-for world, some developers would opt out of Google Play Billing ("GPB") and independently contract with alternative payment processing services does not consider that these payment processing alternatives likely would have been more costly to many developers.

131. According to Plaintiffs, in the but-for world, other payment processors would provide viable billing alternatives to GPB.[145] However, for developers that rely primarily on low-

---

141 Singer Report at ¶¶196, 198. In 2020, according to ONE store, its catalog consisted of 20,000 game titles and 200,000 apps. Moreover, very few developers elected to use their own payment processing systems; according to ONE store, in 2020 – over one year after ONE store reduced its rates – only 30 titles used developer owned payment processing. ONE store Introduction, ONE store Corp. January 2020 at pp. 3, 9, https://dev.onestore.co.kr/devpoc/static-res/files/ONEstoreIntro_dev_en.pdf, accessed November 8, 2021. Indeed, Google Play's response to ONE store apparently was to offer individual developers special terms through its Project Hug and to offer particular consumers special terms through its Play Points program. See e.g., Singer Report at ¶248 ("Google next deployed the Play Points program in response to the ONE store's entry in Korea."); GOOG-PLAY-000953420R – 22,38,40, 60; GOOG-PLAY 000365029; GOOG-PLAY-000005203- 08; GOOG-PLAY-003332070-081.

142 https://www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live

143 Steam, another PC game store, reduced its rates a month prior to the entry of the Epic Game Store. https://www.theverge.com/2018/11/30/18120577/valve-steam-game-marketplace-revenue-split-new-rules-competition.. Even if Steam's rate reductions were in anticipation of Epic Game Store's entry, rates to all Steam developers were not reduced. In fact, Steam reduced rates only to relatively large developers but maintained its 30% rate for developers that generated less than $10 million in consumer spend. Moreover, the Epic Game Store has not been particularly successful. See e.g., https://www.polygon.com/2019/4/5/18295833/epic-games-store-controversy-explained ("Some PC gaming fans are grousing at having to navigate a new store and install new software if they want to play certain games. Some blanch at Epic's comparatively thin store software and how the company is using its *Fortnite* windfall to lock up store-exclusive games. Others have convinced themselves that Epic, its CEO Tim Sweeney, and its Chinese investor Tencent are up to something sinister. Epic itself admits that the storefront launched in a half-baked state."). See Exhibit 28.

144 According to Dr. Singer, ONE store has "overcome the prohibitive restrictions to competition imposed by Google." See Singer Report at ¶198.

145 Consumers identify PayPal, Stripe, and Intuit. Developers identify PayPal, Stripe, Square, and Braintree. See Consumer Complaint at ¶10; Developer Complaint at ¶¶206, 236.

priced app transactions, these alternatives could be more expensive, not less expensive. Table 4 shows that various payment processors' rates include not only a fee based on the percentage of the value of the transaction, but also a fixed charge for each transaction. For example, PayPal, Braintree, Square, and Stripe include a percentage fee plus a fixed charge of $0.15 to $0.49.

Table 4. Fees for Payment Processing Service Providers, 2021

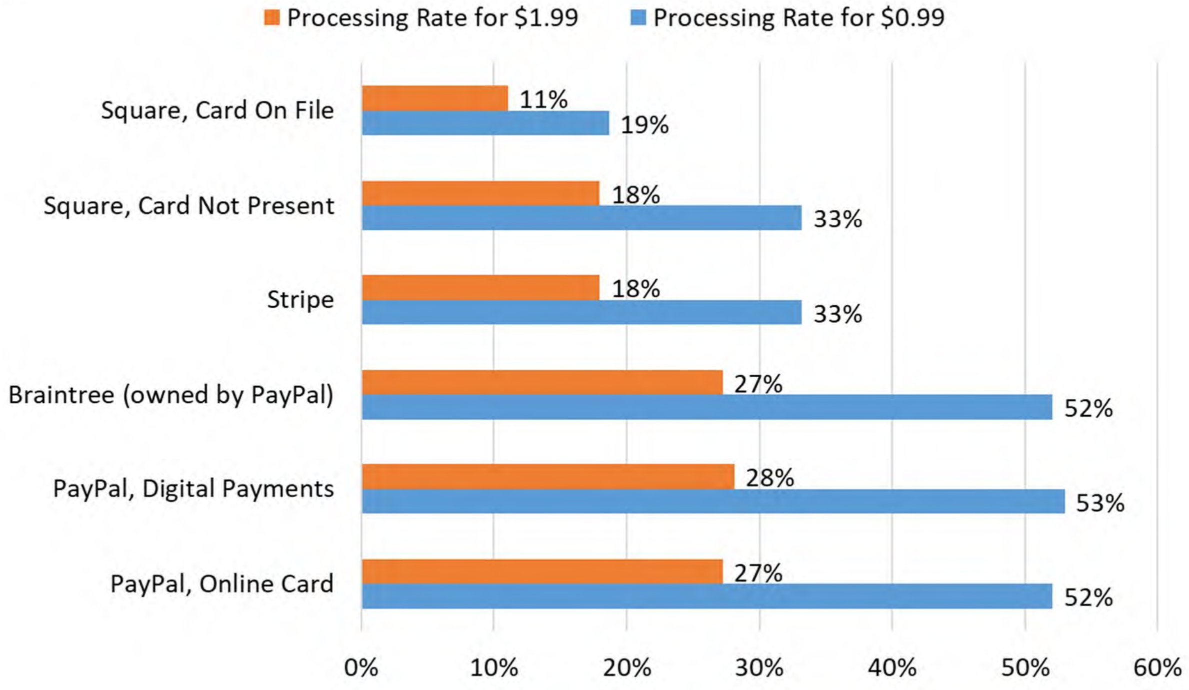
Payment Processing Service Provider	Base Rate (U.S.)
PayPal	2.59% + $0.49 [online card] 3.49% + $0.49 [digital payments]
Braintree (owned by PayPal)	2.59% + $0.49
Stripe	2.9% + $0.30
Square	2.9% + $0.30 [card not present] 3.5% + $0.15 [card on file]

Source:

Exhibit 29.

132. Payment processors' cost to developers can be substantially higher than 30% of the value of the transaction for developers with low prices. For example, a developer with a $0.99 price that used PayPal would pay $0.49 plus 2.59% of the price. The total cost is $0.52 or 52% of the value of the transaction – substantially higher than Google Play's 30% (or 15%) rate. Similarly, a developer with a $1.99 price would pay $0.49 plus 2.59% of $1.99. The total cost is $0.54 or 27%. Figure 5 illustrates that the effective rate of many payment processors for low price points either exceeds or is close to Google Play's 30% (or 15%) rate. Individualized inquiry is required to determine which payment processor a developer would have selected in a but-for world based on its transaction mix and thus to determine costs in a but-for world for each developer.

Figure 5. Effective Payment Processing Rates of Alternative Processors at Low Price Points



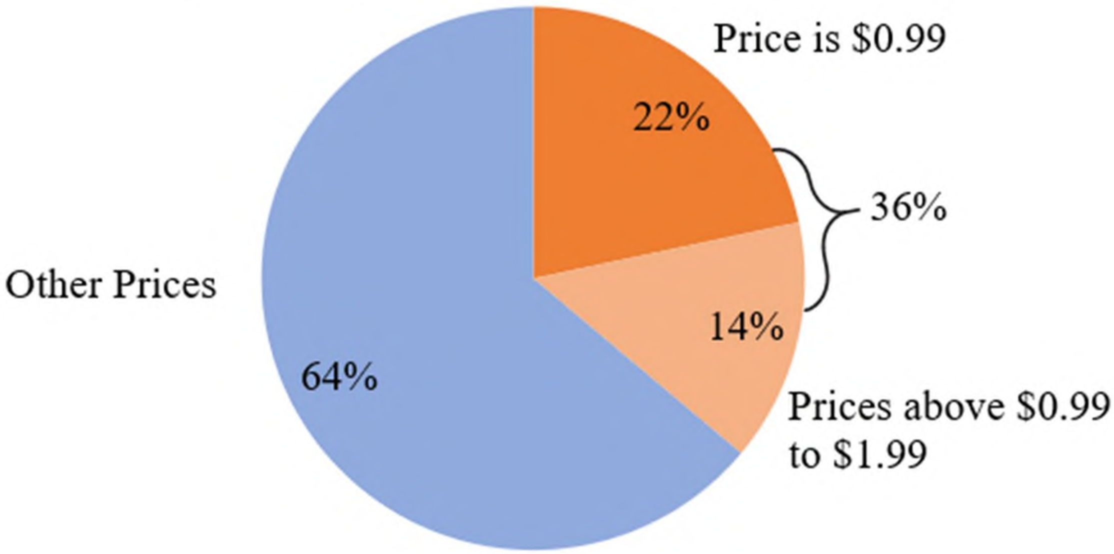
Source:

Exhibit 30.

133. Moreover, a developer that opted out of GPB would additionally have to incur costs for the other services that Google provides, such as customer service, subscription management, and management of billing disputes. The developer would incur the costs associated with integrating those features into any alternative payment processing system, possibly leading to further increases in the developer's costs and reducing its profits. Plaintiffs' experts have not accounted for these costs, which are specific to each app. For $1.99 apps, adding these costs to payment processing costs, could make the developer worse off in a but-for world.

134. A substantial percentage of putative developer class members set prices at $0.99 and $1.99. Figure 6 shows that 22% only sell apps, subscriptions, or IAPs at $0.99, and 36% of putative developer class members sell at $1.99 or lower.

Figure 6. Percentage of Putative Developer Class Members Offering Apps, Subscriptions, and IAPs at Low Price Points, August 2016 – July 3, 2021



Source:

Exhibit 31.

135. Other payment processors likely would not be an economically viable option for developers that sell at (or mostly at) low price points. In the but-for world, those developers likely would not have obtained lower service fee rates by independently contracting with a payment processor, so there would have been no economic reason for Google to reduce its service fee for these developers based on developers' ability to process payments through another processor.[146] Given that those developers did not obtain a lower rate in the but-for world, they would not be impacted and have suffered no injury. In addition, developers that did not pay lower fees would not lower prices, so consumers who purchased from these developers would not be impacted and have suffered no injury. As shown in Figure 6, these potentially uninjured developers constitute between 22% and 36% of the proposed developer class.

136. In addition, because those developers would not have a lower service fee, there is no alleged overcharge to "pass through" to consumers – meaning that members of the proposed consumer class who purchased apps, subscriptions, or IAPs at low price points could not have been impacted, either.

137. Because some developers that sell apps, subscriptions, and IAPs at $0.99 or $1.99 also sell other apps, subscriptions, and IAPs at higher prices, determining whether a developer's service fees would have been lower with alternative payment processors requires analyzing

146 Other alternative payment processors could have higher fees than Google Play. Class representative Pure Sweat Basketball used [redacted]

the relative volumes of sales at the lower price points and the higher price points. For example, in the actual world, a developer that sold 800 IAPs at $0.99 and 200 IAPs at $3.99 would have higher payment processing costs if PayPal processed the developer's payments as compared to Google Play. The developer's payment processing costs would have been the total of its costs of processing the 800 transactions at the $0.99 price and the costs of processing the 200 transactions at the $3.99 price ($531.18) divided the developer's revenue from all of transactions ($1590) – or 33%.[147] Therefore, for developers that sell apps, subscriptions, or IAPs at price points both below and above $1.99, determining antitrust impact would require an individualized analysis for each developer to determine if the developer would have been better or worse off in a but-for world.

B. No Common Proof of Lower Prices for Apps, Subscriptions, and IAPs in the But-For World

138. Whether any individual consumer or developer experienced antitrust impact depends on the prices of apps, subscriptions, and/or IAPs in the but-for world.

139. For consumers, a service fee rate reduction must lead to price reductions to them, otherwise there is no antitrust impact. But even if all service fee rates were lower in the but-for world, it is not the case that all app, subscription, and IAP prices would have been lower. Differences in cost conditions for different apps, differences in the demand elasticities for different apps, and differences in app developers' pricing strategies indicate that many developers would not charge lower retail prices for apps, subscriptions, and IAPs in a but-for world with uniformly lower service fees.

140. If a service fee rate reduction led to lower prices, one must assess the consequences of that lower price for the developer's profits to determine whether the developer experienced an antitrust impact. A developer could reduce its prices in response to a service fee rate reduction or in response to a price reduction by a substitute app. In either case, to determine impact, the effects of the lower retail prices on the developer's profits must be considered.

141. A developer's decision to reduce retail prices when the service fee rate is lower depends on at least three factors: (i) the marginal costs of distributing the developer's apps, (ii) the developer's pricing strategy, and (iii) the developer's price elasticity of demand. Because these factors vary by developer and by app, determining whether a retail price is lower in the but-for world necessarily requires individualized inquiry.[148]

---

147 PayPal's payment processing costs under this scenario would be about 33% of the developer's gross revenue. See Table 4 for PayPal's costs. Applying PayPal's base rate for online credit and debit card transactions, the developer's gross revenue is $1590.00 (= 800 * $0.99 + 200 * $3.99). The payment processing costs are $531.18 = (2.59%*$0.99 + $0.49) * 800 + (2.59% * $3.99 + $0.49) * 200, which is about 33% of gross revenues.

148 See Deposition of Adam Sussman, January 7, 2022, ("Sussman Dep.") at pp. 263-265 [redacted]

1. Developers' Marginal Costs

142. Developers vary in terms of whether their apps have a positive marginal cost or marginal cost close to zero. This is relevant to a determination of impact because the type of economic models that for example, Dr. Singer relies on show that a service fee rate change depends in part on whether the developer's apps have a positive marginal cost or marginal cost close to zero.[149] The models show that if an app has zero (or close to zero) marginal cost, a reduction in the service fee rate will be less likely to lead to a change in the retail price of the app. In economic models of profit-maximization, a developer considers marginal cost and marginal revenue to determine the optimal price and quantity. If marginal cost is zero, profit maximization is equivalent to revenue maximization because costs do not play a role. That is, the profit-maximizing price of the app is also the revenue-maximizing price of the app. Under these conditions, the price that maximizes 70% of revenue will also maximize 85% of revenue (or any percent of revenue).[150] This also applies with regard to subscription prices or IAP prices.

143. For some apps, subscriptions, and IAPs, marginal costs are likely to be zero or close to zero.[151] The cost of producing (e.g., creating) and distributing are incurred for a single consumer to download or purchase, but do not change as more consumers download or purchase it. Thus, the marginal cost of the app, subscription, or IAP is zero. Therefore, a change in the service fee would likely have little or no effect on the price charged to consumers.

144. However, some apps have positive marginal costs. Apps that provide licensed content to consumers may incur incremental costs as more consumers access the content. Music

---

149 Singer Report at ¶225.

150 See e.g., Liobet, Gerard and Jorge Padilla, "The Optimal Scope of the Royalty Base in Patent Licensing," *Journal of Law & Economics*, Vol. 59, No. 1, 2016, pp. 45-73 at 53 ("The case in which $c = 0$ is particularly illuminating of the different effects at work…Under ad valorem royalties, however, there is no conflict between the innovator and the producer, since both firms are interested in maximizing total revenue. Hence, no matter how large s is, the downstream producer always chooses the monopoly price. In other words, there is no pass-through of a higher ad valorem royalty into a higher price."); Kobayashi, Bruce H. and Joshua D. Wright, "What's Next in Apple Inc. v. Pepper? The Indirect-Purchaser Rule and the Economics of Pass-Through," *Cato Supreme Court Review*, 2018, pp. 249-269 at 262 ("To the extent that the marginal costs of producing and distributing another copy of the app is zero, the theoretical calculation of the markup is far from complex—it is simple. And the effect of the Apple 30 percent ad valorem royalty on the optimal price set by the app developer is zero.").

151 Marginal costs of digitally distributed products are sometimes assumed to be zero in the economic literature. See e.g., Weber, Thomas A., "Delayed Multi-Attribute Product Differentiation," *Decision Support Systems*, Vol. 44, 2008, pp. 447-468 at pp. 447, 451 ("Information goods such as computer software or electronic newspapers can be provided by firms at a low marginal cost, though in many cases large capital outlays are required to produce their first unit…. Throughout most of this paper we consider information goods with zero marginal cost, which simplifies the closed-form solutions. … For information goods, the costs of reproduction and distribution are indeed very small, so that this has become a standard assumption in much of the extant literature."); Lambrecht, Anja, et al., "How do firms make money selling digital goods online?" *Marketing Letters*, Vol. 25, No. 3, 2014, pp. 331-341 at 331 ("Such [digital] goods are non-rival, have near zero marginal cost of production and distribution, low marginal cost of consumer search and low transaction costs.").

streaming developers, such as Spotify, may incur the marginal cost associated with licensing fees paid to record labels and music publishers. According to Spotify, it was forced to raise its prices in response to Apple's service fee rates and when it did so, its relative price increased – compared to Apple's price, and the demand for its app service fell.[152] Other types of apps that rely on licensed content and pay licensing fees that depend on the usage of the app would also have positive marginal costs. For example, some games incorporate music and the developers of those games may pay music licensing fees each time a consumer downloads the app and listens to the music.[153]

145. Another example is weather apps that may have positive marginal costs because as more consumers in new locations demand weather-related information, the cost of obtaining the information increases.[154] Some apps may rely on licensed intellectual property and licenses may be based on a per consumer or per transactions basis. Determining which apps have positive marginal cost and which have marginal cost close to zero would require information about the content and the structure of the licensing fees paid by the developers.[155]

146. Other developers may have different types of positive marginal costs.[156] For example, apps that require processing of consumer information (e.g., usernames and communications across consumers) such as multi-player game apps and apps that require relatively high costs for customer support[157] may have positive marginal costs. As the number of app users increases, the processing costs associated with the app increases.

---

152 Testimony of Horacio Gutierrez Head of Global Affairs & Chief Legal Officer, Spotify U.S. Senate Judiciary Committee Subcommittee on Competition Policy, Antitrust, and Consumer Rights April 21, 2021 at fn. 13 ("Spotify could not absorb the IAP tax without raising its prices, because a large component of its costs are the licensing fees paid to record labels and music publishers."); *id.* at p. 8. ("Spotify recognized that it could not compete with Apple's lower price and in May 2016 eliminated IAP, which required it to forgo any in-app purchases of Premium and any in-app upgrades from Free to Premium."). See also European Commission Press Release, Antitrust: Commission sends Statement of Objections to Apple on App Store rules for music streaming providers, Brussels, 30 April 2021 ("Apple charges app developers a 30% commission fee on all subscriptions bought through the mandatory IAP. The Commission's investigation showed that most streaming providers passed this fee on to end users by raising prices.").

153 Choices: The Stories You Play is a top revenue-generating app offered by Pixelberry Studios that incorporates music with its stories. See https://play.google.com/store/apps/details?id=com.pixelberrystudios.choices.

154 See e.g., https://developer.accuweather.com/packages (showing package pricing for Accuweather APIs).

155 See, for example, [redacted] See also Zynga Inc. Form 10-K, 2020 at p. 2 and Electronic Arts Form 10-K, 2020 at pp. 3, 5.

156 See Ghose, Anindya, and Sang Pil Han, "Estimating Demand for Mobile Applications in the New Economy," *Management Science*, Vol. 60, No. 6, 2014, pp. 1470-1488 at 1481 ("Ongoing marginal costs for app developers arise from various maintenance-related tasks after app development.").

157 See https://slack.com/help/requests/new

147. Identifying which apps' marginal costs are positive (and not close to zero) requires information about each app; one cannot assume that all apps have zero marginal cost or that all apps have positive marginal cost. Moreover, identifying the marginal costs associated with an app is not always straightforward. Even developers that evaluate costs associated with their own businesses sometimes have difficulty in separating costs that are fixed from those that vary with output.[158] Individualized information and analysis are required to determine whether even adeveloper that sets prices based on short-term profit-maximization would reduce prices given the claimed lower service fee rate. This requires an app-by-app analysis to determine whether any individual developer or consumer suffered antitrust impact or injury.

2. Developers' Strategy of Setting Prices

i. *Prices that end in "99"*

148. Another factor that explains why determining prices in the but-for world requires an app-by-app inquiry is that developers employ different pricing strategies, including the strategy of setting retail prices that end in "99."

149. Many app, subscription, and IAP prices are set to end in "99," such as $0.99, $1.99, and $2.99. These price points may influence consumers' perceptions of price and thus affect sales.[159] Academic studies suggest that some consumers may pay less attention to the rightmost two digits of a price, so that a price of $0.99 is perceived to be significantly more attractive than $1, even though it is only one cent lower.[160] As shown in Figure 7 over the period from August 2016 to July 3, 2021, 97% of U.S. consumers' retail app transactions were set such that the retail prices ended in "99."

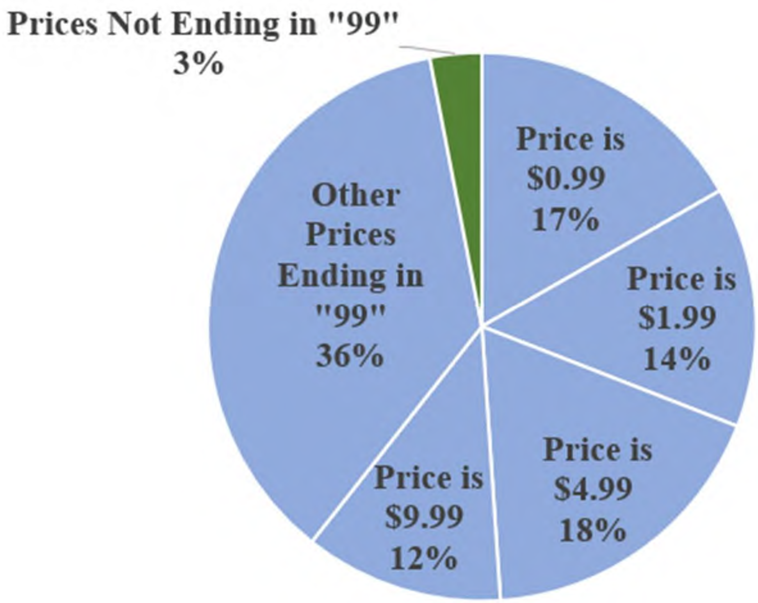
---

158 [redacted]

159 See, for example, Stiving, Mark and Russell S. Winer, "An Empirical Analysis of Price Endings with Scanner Data," *Journal of Consumer Research*, Vol. 24, No. 1, 1997, pp. 57-67 at 57 ("Managers apparently set prices in a manner consistent with the premise that the last digit of a price has a significant impact on sales. Several surveys on what price endings managers actually use have been conducted, and all of these surveys support the premise that firms set prices to appear that they are just below a round number."). See also Schindler, Robert M. and Patrick N. Kirby, "Patterns of Rightmost Digits Used in Advertised Prices: Implications for Nine-Ending Effects," *Journal of Consumer Research*, Vol. 24, No. 2, 1997, pp.192-201 at 193-194; Anderson, Eric and Duncan Simester, "The Role of Price Endings: Why Stores May Sell More at $49 than $44," 2000, at http://ssrn.com/abstract=232542.

160 Bizer, George Y. and Robert M. Schindler, "Direct Evidence of Ending-Digit Drop-Off in Price Information Processing," *Psychology & Marketing,* Vol. 22, No. 10, 2005, pp. 771-783 at 771-772.

Figure 7. Prices that End in "99" as a Percentage of U.S. Consumer Transactions

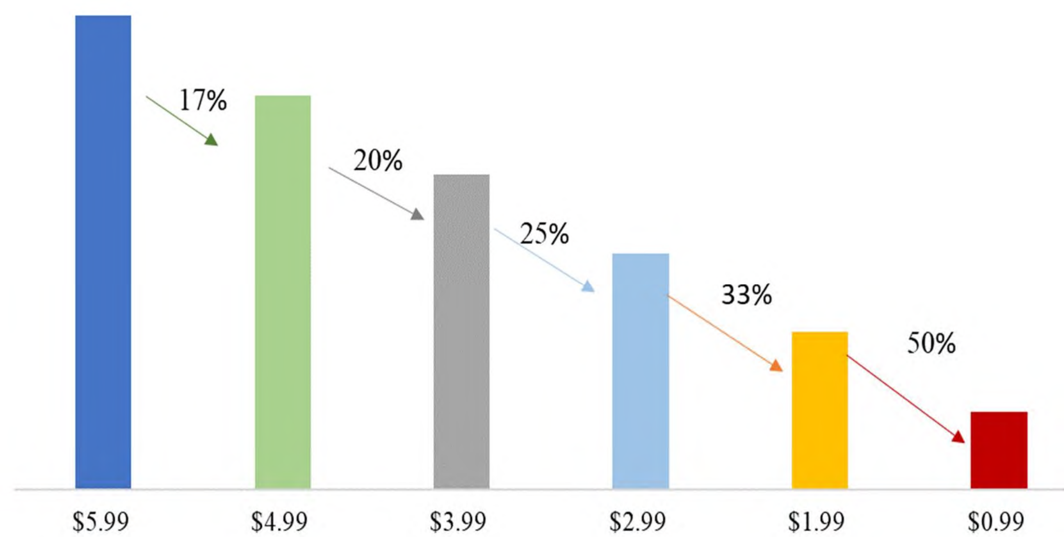


Source:

Exhibit 32.

150. Developers that rely on this strategy to set prices would not reduce prices in response to lower service fee rates if the reduction from one price point to the next would be so large that the developer would lose profits. For example, if a developer's price is initially set at $1.99 and a change in some supply or demand factor would lead to a new profit-maximizing price of $1.63, a developer that sets prices to end in "99" may elect to keep the price at $1.99 instead of reducing the price to $0.99 because profits at $1.99 are higher than profits at $0.99 (and the developer would not want to abandon its pricing strategy and set the price at $1.63). Figure 8 below shows the implied percentage reductions from one price point to another for prices that range from $5.99 to $0.99. A developer with a price of $5.99 would have to reduce prices by 17% to get to the next price point price of $4.99; a developer with a price of $1.99 would have to reduce prices by over 50% to get to the next price point of $0.99. Absent other competitive pressures, a profit-maximizing developer that found it would be worse off by reducing prices from one price point to another would choose not to reduce prices at all.

Figure 8. Percent Reductions from Price Point to Price Point



Source:

Exhibit 33.

151. Many app, subscriptions, and IAP prices exhibit the "stickiness" associated with this strategy. Across all three of these types of transactions, the prices of 73% of apps, subscriptions, and IAPs that appear in the Google Play transactions data for at least one year do not *ever* change.[161] This confirms that developers' reliance on this type of pricing strategy would inhibit price changes in response to service fee rate changes.

152. Given the prevalence and effect of setting prices to end in "99," an individualized assessment of each developer's pricing decisions would be required to understand the actual and but-for prices to determine if prices would have been different in the but-for world and if a consumer was, in fact, injured.

ii. *Developers' Other Price Setting Strategies*

153. Developers use other price-setting strategies, besides short-term profit maximization. For example, Pure Sweat Basketball's decision to lower its monthly subscription price in 2015 from $29.99 to $9.99 was driven by an attempt [redacted][162] Little Hoots selected its price [redacted][163] Rescue Pets sets prices to [redacted]

---

161 See this report's production.

162 [redacted].

163 [redacted]

[redacted][164] Other developers clearly do not operate based on a short-term profit maximization strategy. From the time it initially launched as an app, in 2010, until 2013, Instagram generated no revenue from any source. It was only a year after it was purchased by Facebook that it began to generate any revenue, which it did primarily through advertising.[165]

154. Developers' decisions regarding price changes in the but-for world depend on their pricing strategies. Given that these strategies vary across developers, an individualized analysis of developers is required in order to determine whether a price would have been lower in the but-for world and consumers who purchased those developers apps, subscriptions, or IAPs were impacted.

3. Competitive Conditions Across Apps

155. Another important factor that affects changes in retail prices due to changes in service fee rates is the economic relationships between and among apps. Apps are highly differentiated products and have complex economic interrelationships. Some apps are substitutes and others are complements. If, in response to a service fee rate reduction, some price falls, the demand for competing apps would tend to fall, which would put downward pressure on the prices of those substitute apps. Developers that face more competition will face more pressure to lower prices in response to service fee reductions. Developers that face less competition (from substitute apps) will face less pressure to reduce prices. The extent of competition can be measured by the price elasticity of demand for the app. An app that faces competition from many substitutes will have a more elastic demand than an app without such competition.[166]

156. Determining the price elasticity of demand for an app can be highly complex and individualized because apps are highly differentiated products and there can be many potential substitutes. U.S. consumers made app, subscription, or IAP purchases from 465,245 different apps during the class period.[167] Moreover, consumers download and use many other apps that are free as well as apps that are paid or have subscriptions and IAPs. For example, among six of the consumer class representatives (those that have produced

---

164 [redacted].

165 https://www.investopedia.com/articles/investing/102615/story-instagram-rise-1-photo0sharing-app.asp (Instagram launched in 2010, purchased by Facebook in 2012 for $1 billion in cash and stock); https://www.nytimes.com/2015/06/03/technology/instagram-to-announce-plans-to-expand-advertising.html (first used ads in 2013). Even developers who rely on a profit-maximizing price-setting strategy are different from one another because they rely on different and sometimes multiple sources of revenue. Many developers generate revenue from advertising as well as paid installations, subscriptions, and IAPs. If those developers set prices to maximize short-term profits and generate revenue from advertising as well as paid transactions, a reduced service fee rate may lead to higher retail prices, not lower retail prices.

166 As Mr. Adam Sussman, at Epic, described, [redacted].

167 See this report's production; App-level Spend Data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688.

information), the total number of apps installed or updated between August 2016 to October 2021 ranged from 103 to 297.[168]

157. App differentiation can be observed by the numerous categories of apps. Google Play has 35 categories for non-game apps and 17 different game app subcategories.[169] The categories vary widely; for example, non-game apps include apps for education (over 400,000 apps), business (over 300,000 apps), shopping (over 180,000 apps), and other categories.[170]

158. Within each of the numerous app categories, there are many apps that may be substitutes to varying degrees and apps that may not be substitutes at all. For example, in the Arts & Design app category (which includes over 29,000 apps),[171] Sketchbook, My Drawing, and Tayasui Sketches are all drawing apps, with different features.[172] Those apps, based on their product characteristics, may be potential substitutes for one another. On the other hand, Pixel Studio and U Launcher are two apps also categorized as Arts & Design apps.[173] Pixel Studio, is a pixel art editor for artists and game developers. It has numerous features, including tools that allow the user to create animations, custom brushes, and symmetry drawing. U Launcher, on the other hand, provides customizable home screens for Android phones with various backgrounds, as well as the ability to hide/unhide apps and search apps. Even though these two apps are included in the same category – Arts & Design – they have different purposes and functionality and are directed toward different sets of consumers. As such, Pixel Studio and U Launcher are likely poor substitutes for one another. Google's "Games" category is also broad and includes games for toddlers such as "Thomas and Friends" (described as a "fun, safe & interactive game play for children aged 2-7"),[174] action

---

168 These counts include free apps, paid apps, subscription apps, and apps with IAPs. The counts exclude apps by OEMs and carriers that may have been pre-installed by them and also exclude Google apps. See Exhibit 22.

169 See Exhibit 2 for the list of non-game categories and the percentage of apps in each of the categories. See Exhibit 3 for the list of game categories and the percentage of apps in each of the categories.

170 Exhibit 2.

171 Exhibit 2.

172 Sketchbook, https://play.google.com/store/apps/details?id=com.adsk.sketchbook, accessed March 22, 2022. My Drawing, https://play.google.com/store/apps/details?id=com.raed.sketchbook, accessed March 22, 2022. Tayasui Sketches, https://play.google.com/store/apps/details?id=com.tayasui.sketches, accessed March 22, 2022.

173 Pixel Studio, https://play.google.com/store/apps/details?id=com.PixelStudio, accessed March 22, 2022. U Launcher, https://play.google.com/store/apps/details?id=com.phone.launcher.android, accessed March 22, 2022.

174 Thomas & Friends: Magic Tracks, https://play.google.com/store/apps/details?id=com.budgestudios.googleplay.ThomasAndFriendsMagicalTracks, accessed March 28, 2022.

game "Doom" (described as "Violence, Blood and Gore" for ages 17+),[175] Zynga's card game "Poker – Texas Hold'em,"[176] and home design game "Redecor."[177]

159. Paid apps, subscription apps, and apps with IAPs can be substitutes for free apps. Free apps account for a substantial percentage of apps in all categories.[178] Determining whether the prices of paid apps, subscription apps, or IAPs would have been lower in the but-for world requires analyzing the economic relationships between those apps and the free apps that they compete with, if any.

160. Since different apps face different competitive conditions, an individualized analysis of the competitive pressures on each app is required to assess whether there would have been any changes in the app, subscription, or IAP prices in the but-for world.

C. Economic Reasons Some Members of the Putative Consumer Class Would Not Be Better Off in the But-For World

161. As described above, the proposed class of consumers includes over 92 million U.S. consumer IDs with varying purchase characteristics including, for example, the number of apps purchased (or made purchases from), the number of free apps downloaded, the types of apps purchased and downloaded, the amount of money spent, the prices paid, and the forms of payment used to make purchases, as well as others. Consumer Plaintiffs claim that they would have been better off in the but-for world because the prices they paid for apps, subscriptions, and IAPs would have been lower. However, whether prices would have been lower in the but-for world requires an app-by-app inquiry into (1) whether that developer's service fee rate would have been lower in the but-for world, and (2), if so, whether the developer would pass through that reduction in the service fee rate, to the consumer.

162. There are additional factors that must be considered to determine whether a consumer would have been better off in the but-for world. These factors likewise involve an app-by-app analysis of each consumer's purchases and downloads. As further explained in the sections below, several categories of consumers likely would not be better off in the but-for world, and thus likely would not have experienced antitrust impact.

163. First, some consumers make purchases from developers that set low prices such as $1.99 or less. As discussed above, because those developers likely would not have lower cost options for payment processing in a but-for world, they likely would not have lower service fees for these low-price transactions. Under Consumer Plaintiffs' theory of impact – which posits that developers would respond to lower service fees by reducing retail prices for apps subscriptions, and IAPs – developers that set low prices would have the same service fee rate

---

175 DOOM, https://play.google.com/store/apps/details?id=com.bethsoft.DOOM, accessed March 28, 2022.

176 Zynga Poker- Texas Holdem Game, https://play.google.com/store/apps/details?id=com.zynga.livepoker, accessed March 28, 2022.

177 Redecor - Home Design Game, https://play.google.com/store/apps/details?id=fi.reworks.redecor, accessed March 28, 2022.

178 See Exhibit 2 and Exhibit 3.

and charge the same retail prices in the but-for world as they do in the actual world. U.S. consumers who only made purchases at low prices would not necessarily be impacted.

164. Second, some consumers rely on relatively expensive forms of payment to transact in Google Pay, such as direct carrier billing ("DCB") and gift cards. For example, named Consumer Plaintiff [redacted] used [redacted] to make purchases on Google Play.[179] The cost to Google of processing DCB and gift cards is high relative to other forms of payment.[180] Developers may not find lower cost options for those forms of payment in the but-for world and would either continue to use Google Pay Billing or elect not to offer consumers those payment options. Consumers who prefer and benefit from the availability of those forms of payment either would not be impacted or would have been worse off with respect to those transactions.

165. Third, many consumers download and use free apps. Google Play does not collect service fees from free apps. If Google elects to charge fees on free apps in the but-for world, according to Consumer Plaintiffs' own expert's reasoning, those fees could be passed through to some consumers and consumers who download many free apps would face higher prices on those apps that may or may not offset any reduction in prices on other apps.

166. Fourth, security costs to some consumers likely would be higher due to fewer (or no) security warnings by Google in the but-for world.

167. Fifth, many U.S. consumers paid relatively small amounts for apps, subscriptions, and IAPs in Google Play. Overall, [redacted] putative consumer class members paid less than $5 over the class period. More than [redacted] of putative consumer class members paid less than $50.[181] These relatively low amounts of consumer spend imply that even if the prices paid for apps, subscriptions, or IAPs were lower in the but-for world, many consumers would save relatively little. For a consumer that spent $5, a reduction in service fee from 30% to 15% that is fully passed through in lower retail prices is $0.75. That amount, if it is even realized by the consumer, must be compared to potentially higher costs the consumer would face in the but-for world to determine whether the consumer is impacted. At a minimum, individualized inquiry is needed to determine whether each of the [redacted] consumers (making up [redacted] of the putative consumer class) who spent less than [redacted] over the class period can show antitrust impact and injury.

1. Some Consumers Would Not Have Lower Retail Prices for Apps, Subscriptions, and IAPs in a But-For World

168. Empirical evidence described in this section shows that service fee rate reductions do not uniformly result in price reductions. When reductions in service fees are not passed onto consumers through lower prices, given the Plaintiffs' allegations, those consumers are not harmed by the alleged conduct affecting service fees for those apps.[182] Identifying those

---

179 [redacted]

180 GOOG-PLAY-000337564 at -587.

181 See Exhibit 23.

182 Consumer Plaintiffs attempt to argue that even if prices in the but-for world are not lower, consumers would benefit from "quality improvements in Apps and In-App Content that developers would

consumers that would have paid lower prices in Plaintiffs' but-for world requires an app-by-app, and consumer-by-consumer inquiry.

169. Certain retail prices are the same in Google Play as prices on the relevant developer's website. That is, some developers have chosen not to use different retail prices, even though applicable service fees are potentially different. These developers do not pay a service fee to Google to distribute the content on their own websites, but their retail prices are the same in both channels. For example, the Minecraft game is listed at the same price on Google Play and on the developer's website.[183] A subscription to iHeartMedia has the same retail price on Google Play and the website.[184] A subscription to Pandora Plus has the same retail price on Google Play and the website.[185] This evidence suggests that even if service fee rates on Google Play were lower in the but-for world, some retail prices would not be lower and those consumers who purchased those apps, subscriptions, and IAPs would not be impacted.[186]

170. Those Google Play purchases that would have cost the same if made directly from the developer, where there is no fee to Google, could not have resulted in any loss to the consumer. Moreover, because some developers offer apps, subscriptions, and IAPs through Google Play and their own websites, an individualized analysis of each consumer's transactions is required to determine whether the consumer could have made each of its purchases on a developer's website, and if so, the price available to the consumer from the website.

171. Other developers, in contrast, charge lower prices on their websites than the prices they set on Google Play. For example:

- Pandora Premium offers a subscription to ad-free, personalized music, podcasts, unlimited skips, and make and share playlists, as well as other features. On its website, Pandora offers Pandora Premium for $9.99 per month, with a 60-day free trial period. In Google Play, Pandora Premium is $9.99 per month, with a one-month

---

be able finance [sic] out of monies saved from lower take rates." Singer Report at ¶233. Dr. Singer makes no attempt to identify or quantify any quality improvements and doing so would require a highly individualized analysis into existing and but-for characteristics of apps and app content as well as consumer preferences related to such characteristics to determine whether a consumer values any change in those characteristics that may come about due to such "quality improvements."

183 The Minecraft app's price is $7.49 for Android mobile version on the developer's website https://www.minecraft.net/en-us/store/minecraft-android, accessed March 21,2022, as well as in the Google Play Store, https://play.google.com/store/apps/details?id=com.mojang.minecraftpe, accessed March 21, 2022.

184 https://www.iheart.com/offers/, accessed March 24, 2022. Google Play Store price accessed with an Android device as of March 22, 2022.

185 https://www.pandora.com/plans, accessed March 21, 2022. Google Play Store price accessed with an Android device as of March 22, 2022.

186 One of the consumer class representatives, [redacted]

free trial period. If a consumer subscribes for a year, the average monthly prices of the subscriptions, including the varying lengths of free trial periods are $8.33 per month from the Pandora website and $9.16 from Google Play.[187]

- Bookedin, a scheduling app for businesses, offers subscriptions on its website and through Google Play. On Bookedin's website, the Pro 1 subscription is priced at $29 per month for a monthly subscription or $288 per year (an average price of $24 per month) for an annual subscription. Bookedin offers the same subscription through Google Play priced at $34.99 per month.[188]
- Yoga Buddhi offers subscriptions for its Down Dog app on its website as well as from Google Play. Monthly subscription rates are $7.99 per month on its website and $9.99 per month through Google Play; its annual subscription rates are $39.99 per year and $59.99 per year, on its website and Google Play, respectively.[189] According to Yoga Buddhi, the subscription rates through its website are lower than those through Google Play because Google Play charges "commission fees."[190] Yoga Buddhi's CEO says that it offers consumers the option of purchasing a subscription through its website in addition to the in-app option, and 10% of its Android consumers choose to purchase the subscription through the app at the higher price.[191]

172. These examples demonstrate that some app, subscription, and IAP prices might have been lower in a but-for world if service fee rates were lower (setting aside a developer's pricing strategy and other factors for a moment), but other prices would not have been lower. Determining which prices would be lower and which consumers would be impacted thus requires an app-by-app and consumer-by-consumer inquiry.

---

187 See e.g., https://help.pandora.com/s/article/Royalties-and-Spins?language=en_US. ("Pandora, as of September 2016, has entered into direct licenses with record labels, digital music distributors and music publishers for the rights to play music with semi-interactive features on our ad-supported and Pandora Plus services in the US, and for on-demand rights in Pandora Premium. Pandora pays sound recording royalties in accordance with the terms of its agreements with sound recording owners, collection societies and rights administrators. Artist royalties for the ad-supported service continue to be paid through SoundExchange. Royalties for subscription services are paid through to record labels.").

188 https://bookedin.com/appointment-booking-payment-system-plans-and-pricing/; Bookedin app, https://play.google.com/store/apps/details?id=net.bookedin.bam, accessed March 22, 2022. Google Play Store price accessed with an Android device as of March 22, 2022.

189 https://www.downdogapp.com/purchase; Down Dog app, https://play.google.com/store/apps/details?id=com.downdogapp, accessed March 22, 2022. Google Play Store price accessed with an Android device as of March 22, 2022.

190 https://www.downdogapp.com/faq. According to Yoga Buddhi, its music licensing fees depend on the usage of its apps; therefore its marginal costs are positive. See Epic v. Apple Transcript of Bench Trial Proceedings Day 2 (May 4, 2021), p. 358 (testimony from Benjamin Simon, the founder and CEO of Yoga Buddhi (Down Dog).

191 Epic v. Apple Transcript of Bench Trial Proceedings Day 2 (May 4, 2021), p. 349:5-8; p. 360:8-13; 363:11-25; pp. 365-66 (testimony from Benjamin Simon, the founder and CEO of Yoga Buddhi (Down Dog), stating that 90% of Down Dog's Android users chose to subscribe on the web when the app informed them that prices were lower on the web).

173. I further analyzed price changes in the actual world for apps, subscriptions, and IAPs (collectively, I refer to these as stock-keeping units, or "SKUs"[192]) in Google Play following changes in service fee rates. I found that, among those SKUs, many prices did not change after service fee rate changes took effect.

174. Service fee rate changes during the class period occurred for certain developers and apps due to programs such as LRAP, ADAP, and others; Google Play's 2018 service fee reduction for certain subscriptions;[193] and Google Play's July 2021 service fee reduction pursuant to the Small Developer Program.[194] These service fee rate reductions applied to 22,995 developers worldwide, to 51,847 apps, and to 467,660 SKUs. These SKUs involved transactions by about 53% of U.S. consumers and accounted for 11% of their consumer spend.[195]

175. While service fees changed for the SKUs in my studies, the prices for many SKUs did not change at all during the class period.[196] Over the three sets of SKUs with service fee

[192] Google's transactions data store IAP and app prices by "product ID," which I will refer to as "SKU."

[193] In 2018, Google reduced the service fee rate on subscription IAPs when the consumer's subscription extended beyond a year. See "Google Play Lowers App Subscription Fee to 15 Percent, Matches Apple's Offering," Gadgets 360, October 20, 2017, https://gadgets.ndtv.com/apps/news/google-play-app-subscription-fee-30-percent-to-15-1764923, accessed November 8, 2021. Since different developers and SKUs (e.g., different types of subscriptions) can have a different mix of consumers, the service fee rate associated with a SKU can vary from 30% (no consumer has a subscription more than a year) to 15% (all consumers have subscriptions more than one year). To test whether retail prices would respond to a service fee rate subscription, I look only at SKUs for which the monthly service fee rate for that SKU fell to 20% or lower and remained at that level for at least three consecutive months.

[194] The July 2021 service fee rate reduction applied to a developer's first $1 million in consumer spend. I look at SKUs for which the monthly service fee rate for the corresponding app and purchase type (i.e. paid app downloads and non-subscription IAPs) fell to 20% or lower in at least one month on or after July 2021. See "Changes to Google Play's service fee in 2021," Google, https://support.google.com/googleplay/android-developer/answer/10632485, accessed January 5, 2022.

[195] Exhibit 34.

[196] I used prices net of promotions and discounts in Google Play transaction data from the beginning of the class period to July 3, 2021. See GOOG-PLAY-007203251. Since the transaction data is available for only two days following the July 1, 2021 reduction in service fee rates on the first $1 million of consumer spend, I also used monthly average prices for paid downloads from the App-level spend data for U.S. consumers (GOOG-PLAY-005535886 and GOOG-PLAY-010801688) through December 2021, which correspond with paid download SKUs. A third dataset ("scraped data") was used to analyze paid app download SKU prices following the July 2021 service fee rate reduction. That dataset was created by electronically collecting data for a sample of apps from Google Play once a month, for the paid download prices for those apps. The scrapes were conducted from the U.S., at the beginning of the month starting in April 2021 and collected various attributes of the apps from their publicly available Google Play pages, including their paid download prices. The apps sampled included (1) the top 20 apps in each app category and monetization type based on their 2020 consumer spend; (2) the apps of the top 20 putative developer class members or developers selling to U.S. consumers based on their 2020 spend; (3) the six top 200 charts on Google Play for "free," "grossing," and "paid" apps and for games vs. non-games; and

reductions, 93% exhibited no change in price at all during the class period.[197] That is, among these SKUs, at most 7% responded to the service fee rate change in the actual world. Moreover, among this 7%, some retail prices could have increased rather than decreased and some retail prices likely would have decreased at times that are not relevant to the service fee rate reduction (for example, several months before the rate reduction). However, if the price of SKU associated with a service fee rate reduction is constant throughout the class period – as 93% of these prices are – then these is no reason to assume that the price of that SKU would have responded to a service fee rate change in a but-for world either.

176. I also considered prices of the SKUs one month before and one month after the service fee rate change. I compared prices for paid app downloads, subscriptions, and IAPs before and after the service fee rate change for each SKU.[198] Table 5 shows these results. For subscriptions and IAPs, less than 2% of prices changed after a service fee rate decline. That would imply lack of impact on consumers for many apps. For paid app downloads, between 1% and 13% of prices changed following a reduction in service fee rates. That also suggests limited impact on consumers for many apps. Given how few prices declined following reduction in service fee rates, individualized analysis would be needed to identify which apps, subscriptions, and IAPs prices would have been lower and which consumers would have paid lower prices in a but-for world.

(4) the top 100 free apps in each app category on Sensor Tower that ranks by a combination of factors including active users, downloads, and revenue. The top 100 free app charts by category are from Sensor Tower, https://app.sensortower.com/android/rankings/top/mobile/us/overall. A fourth dataset provided to me by Google showing prices on June 21, 2021, October 16, 2021, and February 6, 2022 ("IAP snapshot data," GOOG-PLAY2-000483364), was used to analyze non-subscription IAP prices following the July 2021 service fee reduction.

197 Exhibit 35.

198 For paid app downloads and subscriptions IAPs I compare prices on June 21, 2021 and October 16, 2021. For non-subscription IAPs, I compare prices on June 21, 2021 and February, 6, 2022. These are prices on these specific days, a snapshot.

Table 5. Summary of Pass-Through Rates Found by Dr. Burtis

Purchase Type	SKUs Total	SKUs Share w/Positive Pass-Through
Paid App Downloads		
Scraped Data	1,557	1%
App-Level Spend Data	8,991	13%
Subscriptions		
Deal Developers	706	2%
Subscription Developers	16,307	1%
IAPs	441,383	2%

Source:

Exhibit 36.

177. The analysis described above suggests that while some retail prices could be lower if service fee rates were lower, other prices would not change. The analysis finds the percentage of SKUs with price reductions following a service fee rate change but does not include a study of the economic factors that would explain whether an observed price change was caused by a service fee rate reduction. As described above, those factors include, at least, the marginal cost of an app, the demand elasticity of the app, and the pricing strategy of the app developer – all of which require an individualized analysis of the app and the app developer. That is, determining which prices would have been lower and which consumers were impacted requires an individualized analysis of those factors and how they affect prices of particular apps, subscriptions, and IAPs. That individualized, app-by-app analysis is necessarily to determine whether a consumer who made purchases of that app was impacted.

178. This analysis indicates that Consumer Plaintiffs cannot show injury to all or nearly consumers through common proof, because Consumer Plaintiffs cannot establish through common proof that all consumers would pay a lower price if service fee rates for developers were lower.

2. Some Consumers Who Transact at Low Price Points Would Not Be Better Off in the But-For World

179. Table 6 shows that 5% of U.S. consumers in the putative consumer class pay only prices of $0.99 and 9% of U.S. consumers pay only prices of $1.99 or lower.

Table 6. Counts of U.S. Consumers in the Putative Class, By Maximum Price Point of Transactions

Category	Count of Consumers	Percent of Consumers	Cumulative Percent of Consumers
Consumers Only Buying at a Max of $0.99	4,868,989	5%	5%
Consumers Only Buying at a Max of $1.99	3,667,004	4%	9%
Consumers Only Buying at a Max of $2.99	4,062,814	4%	14%
Consumers Buying at above $2.99	79,730,461	86%	100%
Total Consumers	**92,329,268**	**100%**	**100%**

Source:

Exhibit 37.

180. As described above, for low price point transactions, a developer's payment processing fee if it chose to independently contract with a payment processor would have been more costly than Google's service fee. These developers could continue to rely on Google Play in the but-for world. With unchanged service fees, there would have been no impact on the developers, and no change in prices to consumers and, therefore, no impact on consumers.

181. These developers could separately contract with payment processors. However, as I discussed above, other payment processors are likely to be more costly than Google Play. If, as Consumer Plaintiffs claim, higher costs lead to higher prices, those developers would raise the prices they charge and the consumers that purchase those apps, subscriptions, and IAPs would have been worse off in the but-for world, at least for those purchases.

182. Determining whether consumers who purchase low price apps, subscriptions, and IAPs are better off in the but-for world must account for either unchanged or potentially higher prices in the but-for world. This requires an app-by-app inquiry to identify such apps. By the same token, identifying consumers who were harmed would require individualized analysis as well. At a minimum, for the 9% of consumers who pay only prices of $1.99 or lower, individualized inquiry is required to determine whether each consumer can even show antitrust impact or injury.

3. Consumers Who Use Direct Carrier Billing Likely Would Not Be Better Off in the But-For World

183. Direct Carrier Billing ("DCB") enables consumers to pay for apps, subscriptions, and IAPs through their mobile phone bill. Approximately [redacted] of U.S. consumers use DCB.[199] The average cost to Google of DCB is [redacted] – as much as [redacted]. The cost of DCB to developers is [redacted] – as much as [redacted].[200] In Consumer Plaintiffs' but-for world, it is likely that Google would have an economic incentive to have varying service fee rates for

199 See Exhibit 38.

200 GOOG-PLAY-000337564 at-587.

different forms of payment or to stop providing particularly costly forms of payments such as DCB.

184. Consumers who rely on DCB as a form of payment for apps or IAPs would not be better off in the but-for world and indeed could be worse off and face higher prices.[201] To determine whether such a consumer was impacted, one would have to evaluate purchases app-by-app to determine whether each purchase would have been subject to a lower (or higher) service fee in the but-for world; whether each purchase would have been subject to a lower (or higher) retail price in the but-for world; and whether, overall, the consumer would have paid more or less money for the same set of purchases in the but-for world.

185. Generally, individualized inquiry is required to determine whether any individual consumer uses DCB and whether that consumer would have been better off or worse off in the but-for world. At a minimum, Consumer Plaintiffs cannot show antitrust impact or injury without individualized inquiry for the approximately [redacted] of U.S. consumers who use DCB.

4. Increased Prices for Free Apps Would Make Some Consumers Worse Off in the But-For World

186. Many consumers install free apps, including both free apps with ads and free apps that do not have ads. In the but-for world, Google could elect to charge service fees for free apps. Dr. Singer's opinion (which I have criticized below) that nearly all fee reductions would have been passed through to consumers, suggests that any fee increases for free apps would also be passed on to consumers in the form of higher prices. Thus, even accepting Dr. Singer's opinion that consumers were injured by a supracompetitive service fee, members of the putative consumer class who also download free apps could be worse off in the but-for world if increased prices for free apps are greater than any decrease in prices of paid apps, subscription apps, and apps with IAPs.

187. Google has considered alternative monetization strategies for Google Play that, if adopted in the but-for world, would lead it to charge service fees for free apps. The effect of alternative monetization strategies for Google Play must be considered in evaluating consumer impact. As I understand Plaintiffs' but-for world, Google Play would not collect any service fees from developers who used different payment processors. Given that a considerable portion of Google Play's revenues are at risk in Plaintiffs' but-for world and Google would continue to incur operating costs to provide developers with the services and benefits in Google Play, it would be economically rational for Google to change its monetization strategy.

188. Google has considered alternative ways to monetize Google Play. One of the alternatives considered is a fee per download.[202] Because about 87% of all Google Play developers offer only free apps, a fee per download strategy would mean that 87% of all Google Play developers would pay service fees for the first time in a but-for world.[203] If Consumer

---

201 [redacted], one of the consumer class representatives, used DCB to make purchases on Google Play. [redacted]

202 See GOOG-PLAY-006990552. (A fee of [redacted] to [redacted] was considered with some level of [redacted] allowed.)

203 See Exhibit 7.

Plaintiffs' theory that such costs would be passed through to consumers were right, adding a service fee for developers of apps that are free today would result in higher prices for free apps. As noted, developers who set retail prices to maximize profits do so by considering marginal revenue and marginal cost. Since a fee per download is a per unit marginal cost, imposing such a fee on developers of free apps would tend to cause their marginal costs to increase and in turn tend to cause retail prices to rise.[204]

189. For these reasons, determining impact for an individual consumer must involve an analysis of the consumer's free apps. For example, one of the consumer class representatives, [redacted] spent a total of [redacted].[205] Even if the prices of all of the apps, subscriptions, and IAPs purchased by [redacted] were allegedly inflated by 15%, the total overcharge for [redacted] would be [redacted]. If only half of the free apps downloaded by [redacted] became paid apps and were priced at only $0.99, [redacted] would be worse off in the but-for world, paying more than in the actual world.[206] Determining impact for any consumer must consider not only whether the app, subscription, and IAP prices would be lower in the but-for world, but also whether the developers of free apps downloaded by that consumer would decide to charge for those free apps. This requires individualized app-by-app and consumer-by-consumer analyses.

5. Higher Security Costs for Some Consumers in the But-For World

190. Consumer Plaintiffs' allegation that Google's security warnings and other measures related to security lessen competition does not consider that many consumers value security protection and could be worse off in a but-for world without these protections.[207]

191. Some consumers pay relatively substantial amounts to ensure that their mobile devices and personal information remain secure.[208] Such expenditures are consistent with

---

204 A fee per download not only would raise costs to developers of free apps but because the fee is a per unit cost to the developer, those costs are more likely to be passed through to consumers. That is, even if the other marginal costs of the developer are zero, a per unit service fee will increase the developer's marginal costs and lead to higher prices in a model in which developers maximize profits (setting aside other issues, such as the demand elasticity and the developer's pricing strategy).

205 Exhibit 22.

206 If [redacted] of the [redacted] free apps became paid apps and were priced at $0.99, Ms. [redacted] would pay [redacted] for the apps that had been free. The hypothetical overcharge would have been 15% of [redacted], or [redacted]. See Exhibit 22.

207 Consumer Complaint at ¶43 ("Notably, Google tries to deter sideloading through an array of technological hurdles including a complicated multi-step process requiring the user to make changes to the device's default settings and manually granting various permissions, while encountering multiple, unfounded security warnings that suggest sideloading is unsafe.").

208 It is a basic economic concept that consumers acquire products based on choices that involve the weighing of values and costs. See, for example, Barron, John M. and Gerald J. Lynch, *Economics*, Third Edition, 1992, at pp. 33-39. See also Samuelson, Paul A., "Consumption Theory in Terms of Revealed Preference." *Economica*, Vol. 15, No. 60, 1948, pp. 243-253 at 243 ("the individual guinea-pig, by his market behaviour, reveals his preference pattern – if there is such a consistent pattern."). The fact that some consumers make purchases to avoid security problems reflects the value, that those consumers place on security relative to the cost of obtaining the security.

consumers' valuing the security of their private information.[209] If Google is forced to reduce or eliminate warnings or other conduct related to security, then some consumers likely would find it necessary to spend more on security.

192. Security apps can be expensive and could easily be more expensive than the reduced retail prices for consumers who spend relatively low amounts on Google Play. These types of differences between the actual and but-for world should be considered in determining whether a consumer is better off in Plaintiffs' but-for world.

193. Class representatives install and purchase security and privacy apps.[210] For example, [redacted][212] These class representatives' purchases demonstrate that they were willing to pay relatively high amounts for privacy or security.

194. There are numerous security and privacy apps available, and they are purchased by many consumers.[213] A search of "security apps" in Google Play (e.g., entering "security app" into the Google Play search bar) identifies over 200 different apps. Among the top ten security apps available on Google Play, ranked by U.S. consumer spend, there are two apps that provide security protection by scanning incoming files for malware – [redacted] and [redacted] [redacted] has more than [redacted] consumer installations and [redacted] has more than [redacted] installations. Both apps are free to download but offer subscriptions that range from $0.99 per month ($11.88 per year) to $89.99 per year for [redacted] and $0.99 per month ($11.88 per year) to $104.99 per year for [redacted].[214] These subscription amounts are high relative to the amounts many U.S. consumers spend in Google Play over the more than five-year class period: as Table 3 above shows, [redacted] putative

209 The terms "security" and "privacy" are related and can refer to similar issues. See e.g., https://www.ftc.gov/system/files/documents/reports/privacy-data-security-update-2019/2019-privacy-data-security-report-508.pdf.

210 See Exhibit 39.

211 [redacted]

212 [redacted]

213 McAfee, an on-line protection company and developer of security apps describes internet security as consisting of "a range of security tactics for protecting activities and transactions conducted online over the internet. These tactics are meant to safeguard users from threats such as hacking into computer systems, email addresses, or websites; malicious software that can infect and inherently damage systems; and identity theft by hackers who steal personal data such as bank account information and credit card numbers. Internet security is a specific aspect of broader concepts such as cybersecurity and computer security, being focused on the specific threats and vulnerabilities of online access and use of the internet." See "What Is Internet Security?" McAfee, https://www.mcafee.com/enterprise/en-us/security-awareness/cybersecurity/what-is-internet-security.html, accessed November 18, 2021.

214 See Exhibit 40.

consumer class members paid less than $5 over the class period and [redacted] of putative consumer class members paid less than $50.

195. In Plaintiffs' but-for world, where Google does not engage in the challenged conduct related to security, some consumers likely would purchase more security and privacy protection. Other consumers might not change spending but could face a higher probability of experiencing a security breach. Determining whether a consumer is impacted must consider not only whether developers' retail prices are lower, but also whether the costs of privacy and security are higher. As shown above, the cost of security can be substantial. At the same time, many U.S. consumers spend relatively low amounts in Google Play. For example, as Table 3 above shows, [redacted] of U.S. consumers spent less than $10 over the class period.[215] Even if one of those consumers experienced an overcharge of [redacted], that overcharge ([redacted]) would have been less than security apps that are [redacted] per month [redacted] per year and more than [redacted] over the class period.[216]

196. Individualized inquiry is required to determine whether any individual consumer would pay more money for security in the but-for world if Google's security warnings were removed. For those consumers who would pay more for security in the but-for world, further individualized inquiry is required to determine whether that consumer would be better off or worse off overall in the but-for world, and thus whether that consumer can show antitrust impact and injury. At a minimum, this individualized inquiry is necessary for the [redacted] of U.S. consumers who spent less than $10 over the class period.

D. Different Members of the Putative Developer Class Likely Would Be Better Off Or Worse Off in a But-For World Depending on How Google Play Changes its Monetization Strategy

197. In determining whether a putative developer class member was impacted, one must consider that Google would have a strong economic incentive to change the way it monetizes Google Play in Plaintiffs' but-for world, especially since Google Play incurs significant fixed costs to provide valuable services and benefits to developers and consumers. In Plaintiffs' but-for world, if one accepts their assumption about lower across-the-board service fees, Google Play would have less revenue due to the lower service fee rate and, presumably, reduced transactions due to the entry and expansion by competitors. Additionally, it is my understanding that Developer Plaintiffs claim that if a developer chose to separately contract for payment processing, Google would not earn any revenue from the developer. Service fees generated from subscriptions and IAPs accounted for 99% of total Google Play revenue during the class period.[217] Therefore, in the but-for world described by Developer Plaintiffs,

---

215 See Exhibit 23.

216 About [redacted] of U.S. consumers spent less than $100 over the class period. A 15% overcharge to these consumers implies a $15 overcharge which is still lower than the costs of security over the class period. See Exhibit 23.

217 See Exhibit 4. Plaintiffs' experts acknowledge that Google Play incurs significant fixed costs but fail to take into account that in a but-for world in which Google has fewer transactions, its average fixed costs would increase and its average margin, taking into account those fixed costs, would fall. See Singer Report at ¶262 and Sibley Report at ¶245.

nearly all of Google Play revenues would have been "at-risk."[218] Those revenues, of course, provide a means for Google to support Google Play and invest in services and benefits to developers. Faced with that risk, Google would have an incentive to find some other mechanism to generate revenue for Google Play.[219] Any consideration of antitrust impact must account for potential changes to Google Play's business model in a but-for world. The fact that Google has considered a range of different business models for Google Play in recent years underscores that such changes are at least plausible and should be considered in determining impact. Google has considered several ways to adapt to the changed competitive circumstances Plaintiffs describe and still achieve its objectives of maintaining a high-quality app store that provides value to both developers and consumers while at the same time generating revenue for itself.[220] Different alternative business models would affect some developers differently. Below I discuss certain of these strategy changes and their implications for different sets of developers. Without exception, if implemented, the changes Google has considered would have caused some developers to be better off and others to have been worse off. These changes would also have implications for consumers, leaving some consumers better off and some worse off.

1. Fee Per Download

198. Google has considered charging a [redacted].[221] Such a strategy would mean that about [redacted] of all Google Play developers[222] (those who offer [redacted]) would, for the first time, be subject to [redacted]. The approximately 30% of putative developer class members that also distribute [redacted][223] would, for the first time, have [redacted] that they distribute.

199. Depending on the [redacted], other features of Google Play and the number of a developer's downloads, a particular putative developer class member may be worse off with [redacted] than with Google's current service fee rate. For example, [redacted]

---

218 As described below, Consumer Plaintiffs' expert, Dr. Singer, asserts that Google Play's share of IAP transactions would decline from about 97% to 60% — a nearly 40% reduction in share. Dr. Singer has no basis for his assumption, but it serves to illustrate the potential loss to Google Play in the but-for world and the clear rationale for Google to respond.

219 See Deposition of Paul Feng, January 14 and 18, 2022, ("Feng Dep.") at pp. 158-159 ("If users are led to a way to pay outside of Play billing today, we would not be able to collect a service fee. I think if that was the main mechanism that purchases are made through Play-distributed apps, it's likely we would find mechanisms to charge a service fee."). [redacted].

220 Google considered numerous alternative strategies. See, for example, GOOG-PLAY-000561051.R (May 2019), GOOG-PLAY-003331592.R (August 2019); GOOG-PLAY-000444214.R (September 2019); GOOG-PLAY-003335786.R (August 19, 2020).

221 See GOOG-PLAY-000336574 at 588; GOOG-PLAY-006990552 at 554, 565 (Google considered a fee of [redacted] to [redacted], with some level of [redacted] allowed).

222 Exhibit 7.

223 Exhibit 41.

[redacted] offers an app with over [redacted] downloads and consumer spend of about [redacted] over the class period.[224] At a 30% service fee rate, this putative developer class member's fees would be approximately [redacted]. However, if service fees were assessed at a [redacted]' service fees would have been about [redacted][225] Depending on the number of downloads and the amount of consumer spend, some developers would have been better off with [redacted] but others would have been worse off.

200. If, in the but-for world, Google had changed its monetization strategy to [redacted], a putative developer class member with free apps would have service fees on the download of free apps that it does not in the actual world. Determining whether a putative developer class member is better off in the but-for world would require analyzing not only the putative developer class member's apps that generate revenue through consumer spend, but also its free apps.[226]

2. App or Developer [redacted]

201. Google has also considered charging developers fees for [redacted][227] In the actual world, the only fee Google charges developers

---

[224] https://play.google.com/store/apps/developer?id=Lowtech+Studios, accessed January 24, 2022; see this report's production.

[225] One strategy considered by Google was to charge [redacted] See GOOG-PLAY-000336574 at 588; GOOG-PLAY-006990552 at 554, 565. For [redacted], that would imply [redacted].

[226] Consumer Plaintiffs' expert, Dr. Singer argues that Google would not charge a fee per download in the but-for world because it would discourage consumers from using Google Play and thereby reduce the overall value of the platform. He also claims that it is in Google's interest, more broadly, to encourage consumers to use Google Play because it generates revenue from advertising. See Singer Report at ¶¶270-273. Every monetization strategy, however, has certain costs and benefits to Google, consumers, and developers. Google Play's existing strategy provides certain benefits and efficiencies. However, the strategy also leads to a relatively small number of large developers providing a substantial portion of overall fees which, arguably has other, detrimental, consequences to Google Play. Google must balance these considerations in choosing a method to generate revenues and fund the operation of the store. In a but-for world where its existing strategy is no longer providing Google with the ability to provide a high-quality app store, it will be forced to consider other strategies even if those alternative strategies have certain costs. See e.g., Feng Dep. at pp. 353-355; GOOG-PLAY-003335685.

[227] Apple charges developers $99 per year. See https://developer.apple.com/support/enrollment/.

Also, Developer Plaintiffs have suggested that Google should charge free apps a "reasonable fee" for the distribution of apps in Google Play. See Developer Complaint ¶ 210 ("Nor is the tie necessary to prevent "free riding" by developers as to distribution via the Google Play Store. In fact, Google's current model encourages free riding. Among the apps that benefit from being on the Google Play Store but do not sell digital goods are many categories of very valuable commercial apps such as, for example, those used by banks and other financial institutions, brokerages, insurance companies, and real estate services to interact with customers, in addition to those apps that sell billions of dollars of physical goods (e.g., Amazon), services (e.g., Uber), or advertising (e.g., Facebook). Google could elect to charge a reasonable fee for the Google Play Store's distribution services, but it does not. Instead, it reaps a monopolistic windfall from

to publish apps in Google Play is a one-time fee of $25.[228] Google has considered adding [redacted]. For example, Google considered several [redacted] The [redacted].[229]

202. Many proposed developer class members would be worse off in a but-for world in which Google charged [redacted] As shown in Figure 9 below, during the class period, more than [redacted] of the putative developer class members paid less than [redacted]. Service fees for named Developer Plaintiff Peekya App Services were [redacted] during the class period through 2021.[230] If, in the but-for world, Google Play charged these developers [redacted] instead of the 30% service fee rate, more than [redacted] of these developers' fees would have been lower in the actual world than in the but-for world.

Android in-app payments, to the detriment of developers and consumers alike."). See also Complaint, *State of Utah et al. v. Google LLC*, filed July 7, 2021, Case No. 3:21-cv-05227 ("Plaintiff States' Complaint") ¶ 189.

228 https://support.google.com/googleplay/android-developer/answer/6112435.

229 See GOOG-PLAY-006990552 at 554 and GOOG-PLAY-000565001.R at 019.R. Other amounts considered were a [redacted]. Google also considered [redacted] Apple charges developers who list apps in its App Stores a fee of $99 per year. See https://developer.apple.com/support/compare-memberships/.

230 See this report's production.

[redacted]

3. [redacted] Imply Not All Service Fee Rates Would Have Been Lower in the But-For World

203. Documents produced in this case show that Google also considered introducing [redacted] whereby service fee rates would [redacted].[231] If Google implemented [redacted] in the but-for world, some developers would not have been impacted.

204. One proposal that Google considered was to reduce the service fee rate based on when a consumer [redacted] For example, Google considered having a 30% rate for transactions associated with an individual consumer [redacted] but having a [redacted] rate apply [redacted].[232] The distribution of the

---

231 See e.g., GOOG-PLAY-006990552 at 554 [redacted]; GOOG-PLAY-003335786.R at 808.R [redacted] See also GOOG-PLAY-003331592.R at 605.R; GOOG-PLAY-000565001.R at 002.R.

232 GOOG-PLAY-003331592.R at 606.R shows different policy parameters and the resulting average effective service fee rates, ranging from [redacted] to [redacted]. In 2020, a policy based on [redacted] for the

[redacted] would determine the overall service fee rate. If the distribution of a developer's users were skewed toward [redacted] the developer's overall service fee rate would tend toward [redacted]. On the other hand, if the distribution of a [redacted], the developer's overall service fee rate would tend toward [redacted]

205. Many apps generate transactions for [redacted] If Google adopted this [redacted] all the users of those apps would make purchases from the app for [redacted] and all transactions for those apps would have been subject to a 30% service fee rate. Among putative developer class members' apps that were published between 2016 and 2020, [redacted] existed less than one year,[233] as seen in Table 7. Thus, for those [redacted] of apps, developers were no worse off in the real world than they would have been in a but-for world with [redacted]

Table 7. Putative Developer Class Members' Apps with Lifespan of 12 Months or Less

First Publish Year	Putative Developer Class Members' Apps	Putative Developer Class Members' Apps with Lifespan of 12 Months or Less	Percentage
2016	14,558	4,069	28%
2017	14,899	6,275	42%
2018	11,079	4,850	44%
2019	10,220	4,607	45%
2020	9,891	5,315	54%
2016 - 2020	60,647	25,116	41%

Source:

Exhibit 42.

206. Google also has considered lowering the service fee rate for those apps that [redacted] Several ways of implementing this strategy were considered, including different service fee rates depending on [redacted] For example, one structure that Google considered was that a service fee rate of [redacted] would apply to transactions generated from [redacted]; and a service fee rate of [redacted] would apply to transactions [redacted]ative monetization strategy would have differential effects across developers depending on how each developer [redacted]

first year and [redacted] thereafter was recommended in a policy review. See GOOG-PLAY-006990552 at 554.

233 In the data, these are apps with recorded transactions for a period of 12 months or less.

[redacted][234] Of course, large, well-known developers that have recognized app names and can afford their own advertising (through, e.g., "deep links") are more likely to obtain more users through developer-owned sources. Therefore, this [redacted] monetization strategy would have benefited those large, well-known developers, while not necessarily benefiting the small or start-up developers that account for a substantial percentage of the proposed developer class members.[235] Individualized inquiry would be required to determine how any developer would have been affected by this kind of service fee and whether that developer could show antitrust impact or injury.

4. Lower Service Fee Rates Accompanied by [redacted]

207. As part of Google's consideration of alternative business models for Google Play, it considered lowering service fee rates but [redacted].[236] For example, Google considered that it could increase the [redacted] in Google Play.[237]

208. Some developers would have been better off if Google had adopted such a strategy. For developers that value and are willing to pay for user acquisition and re-engagement, the increased ability to [redacted] would have been an advantage. Those developers would have been able to obtain the services they value in the amount that they allowed.

209. However, increasing the [redacted], even with lower service fee rates, would not have benefited developers that rely on [redacted], as many small developers do.[238] [redacted] or other mechanisms certain developers use to obtain consumers.

210. A strategy for monetizing Google Play that relied more on [redacted] would have been more costly for developers that [redacted] but pay relatively small

---

234 See e.g., GOOG-PLAY-003335786.R at 808.R; GOOG-PLAY-0110236932.

235 It also contemplated a strategy in which these two components would be combined and that a lower service fee rate would be applied for all transactions [redacted]. These alternatives would favor the relatively small number of large, well-known developers included in the proposed developer class and not the substantial percentage of smaller developers.

236 See GOOG-PLAY-006990552 at 554, GOOG-PLAY-003335786.R at 808.R, GOOG-PLAY-003335685 at 694, and GOOG-PLAY-000561051.R at 061.R.

237 See GOOG-PLAY-003335786.R at 808.R and GOOG-PLAY-000561051.R at 061.R.

238 See GOOG-PLAY-003335786.R at 808.R ("Business models we have considered [include] [redacted] Linked to value; UX deterioration; Difficult for small partners.").

amounts of fees (in dollars), even at lower service fee rates.[239] Some developers have relatively low amounts of consumer spend and therefore have relatively low amounts of service fees – even at a 30% service fee rate. For those developers, it is likely that [redacted] would exceed those service fees.

211. Even for some relatively large developers, [redacted] could have exceeded the amount of the service fees associated with their apps. [redacted][240]

212. Antitrust impact must consider that Google would have strong economic incentive to change the way it monetizes Google Play in Plaintiffs' but-for world. Moreover, Google has considered different monetization alternatives in recent years, making such changes plausible. Importantly, any one of the changes Google has considered would have caused some developers to be better off and others to have been worse off in the but-for world. Determining which developers would have been better off in the but-for world varies depending on the monetization strategy Google would have made in the but-for world as well as the characteristics of individual apps and developers.

E. Additional Economic Reasons Some Members of the Putative Developer Class Would Not Be Better Off in the But-For World

213. Determining whether a putative developer class member is better off in the but-for world requires a consideration of the developer's profits, which in turn requires consideration of the developer's service fee rate, prices, and costs in the actual and but-for worlds.

214. As described above, not all service fee rates necessarily would have been lower in the but-for world and it is not necessarily the case that service fees would have been assessed as a percentage of consumer spend. But even if service fee rates would have been based on a percentage of consumer spend and a developer's service fee rate was lower, the developer may still have earned lower profits because the developer's revenues would have been lower or its costs would have been higher in the but-for world.

1. Some Developers Would Have Incurred Higher App Distribution Costs in the But-For World

215. Developer Plaintiffs claim that in the but-for world, there would have been more alternative app distribution channels.[241] Unless all consumers would have utilized all distribution channels (and did so with at least a similar amount of intensity) then developers

---

239 See GOOG-PLAY-003335685 at slide 10. One proposal, described as a "strawman," was to increase the [redacted] Others listed on the same document included reducing the service fee rate to [redacted]. See also GOOG-PLAY-003335786.R at 814.R.

240 See GOOG-PLAY-006990552 at 554 [redacted] work because devs value UA and re-engagement and are willing to pay for it.").

241 See, for example, Sibley Report at ¶211. Dr. Sibley also concludes that the alleged conduct led to barriers to entry. See Sibley Report at ¶¶175, 202, 207.

would have needed to utilize more app stores than they do in the actual world to reach the same number of consumers or forego distribution to some subset of consumers. That is, in the actual world, consumers utilize a certain number of app stores, including Google Play. In the but-for world, with more app stores, even assuming an increase in competition, consumer demand would have been more spread across more app stores and so, to reach the same number of consumers, developers would have had to offer their apps on more stores.

216. In addition, according to Developer Plaintiffs, there would not only have been more stores in the but-for world, but those stores could have used different "forked" versions of Android.[242] Different "forked" Android operating systems imply that the technical requirements for developers to list apps on stores developed for each fork are different from – that is, mutually incompatible with – one another. Depending on the extent of the differences across variants of Android and app stores, a developer listing an app in multiple stores or models of Android incompatible with one another could have been forced to create different versions of the app (and different versions of each update) for each store.

217. For some developers, distributing through more stores or stores that have varying technical requirements would have increased app distribution costs.[243] For example, when Epic Games Inc. was developing Fortnite for Android, because there were different versions of Android and types of Android devices (implicating different device specifications, screen sizes, etc.), [redacted][244] In a but-for world with even more and more varied platforms, developers like Epic Games would have had to work with even more different platform teams to resolve technical issues and ensure their apps worked well for consumers.

---

[242] See, for example, Developer Complaint at ¶¶71-72, Sibley Report at ¶¶222-225. Dr. Sibley refers to Google's contracts with OEMs that "restrict OEMs' ability to manufacture and sell any device that uses a version of Android that is not approved by Google."

[243] See, for example, Deposition of Lawrence Koh, December 9, 2021, ("Koh Dep.") at pp. 30, 36, 134, 321-22 (describing that the decision to list an app on multiple stores requires a consideration of the costs and benefits associated with doing so). See also https://developer.amazon.com/docs/app-submission/migrate-existing-app.html (documentation regarding Migrating an Existing App to the Amazon Appstore) and GOOG-PLAY-000560564 at 575 ("So far only 20 of top 100 apps on Play have published to Amazon. Many (like [redacted] are concerned about the large and ongoing development / maintenance burden and potential loss of Go Global support. Decisions may change as Amazon gets more users.").

[244] See Deposition of Christopher Babcock, February 17, 2022, [redacted]

218. Characteristics of an app, as well as characteristics of consumer demand for that app, will affect the degree to which the developer would have been subject to increased distribution costs in the but-for world. For instance, technical complexity of the app and the frequency of updates, affect the difficulty of adapting an app to different platforms and thus affect the amount of increased distribution costs. In addition, those apps would have needed to be monitored more closely, on more platforms, to ensure the app is working properly and satisfies consumer expectations.

219. There is substantial variation across apps in terms of their technical complexity. Apps that perform relatively complex functions, have relatively more features, or work differently on different devices may be more difficult and costly to adapt to different platforms. Simple apps, on the other hand, will generally be easier and less costly to adapt. Some measures of app complexity used in the literature include the size of the app and the version of the Android operating system required for the app to perform among other characteristics.[245]

220. Some apps that require more frequent updates would have incurred higher distribution costs in the but-for world, because the apps would have needed to be updated in multiple stores on multiple different platforms. Updates are performed both because the app must be revised to "fix bugs" or to upgrade security as well as because updates provide improvements such as new features or enhancements.

221. Developers' costs in the but-for world also may have been higher for other reasons. There are costs associated with transacting with an app store, including, for example, the costs of understanding and complying with each app store's terms and conditions, and marketing the app in each store. As a developer offers an app in more stores, those costs can increase.[246]

222. Characteristics of consumer demand for an app would likewise affect distribution costs in the but-for world. If the app's value increases with the number of users – as the value of dating apps, social network apps, and multi-player gaming apps, for example, do – the success of the app will require broad distribution across multiple stores and the developer

---

245 See, for example, Ghose, Anindya, and Sang Pil Han, "Estimating Demand for Mobile Applications in the New Economy," Management Science, Vol. 60, No. 6, 2014, pp. 1470-1488. According to these criteria, a relatively simple app, with relatively few features is KLWP Live Wallpaper Pro Key, an app categorized as a "Tool," published by Kustom Industries. The size of the app is 1.8 megabytes; it requires Android version 4.4 or higher. An example of a relatively complex app is Five Nights at Freddy's AR: Special Delivery, an app categorized as a Strategy Game, published by Illuminix Inc. The size of this app is 185 megabytes; it requires Android version 7.0 or higher. https://play.google.com/store/apps/details?id=org.kustom.wallpaper.pro; https://play.google.com/store/apps/details?id=com.illumix.fnafar.

246 See [redacted]

will have strong incentives to incur additional distribution costs. Some developers require broad distribution to attract the few paying consumers that account for a substantial amount of overall paid transaction revenue. Many apps that are free to download, for example, have a high number of users but derive revenue from IAP transactions from a relatively small number of those users. Zynga Inc. is a developer of apps that are free to download. Its paying consumers account for about 3% of the total number of consumers that download the app.[247] Zynga needs broad distribution of its app to find the customers that generate revenue.

223. Listing an app in multiple stores can also make discovery of the app more costly. The success of some apps for some developers is driven, at least in part, by obtaining a given number of downloads in a store[248] or reaching the "top" of an app category.[249] The benefits from the discovery associated with such featuring will be more difficult to obtain if the app must split its downloads across multiple stores.

224. Variation across developers in the factors that affect distribution costs in the but-for world implies that determination of impact will vary across developers, requiring individualized inquiry for each developer.

2. Some Developers Likely Would Have Obtained Fewer App Distribution Services in the But-For World

225. Google provides developers with numerous valuable services.[250] In a but-for world, Google could have adjusted the type and/or level of services provided to developers. Changes in the services offered or the terms under which the services were offered would have had a differential impact on developers depending on how valuable a particular service is to them and whether they could have afforded to purchase the service from Google or elsewhere.[251]

---

247 Zynga Inc. Form 10-K, 2020, at p. 3.

248 See, for example, [redacted]

249 See, e.g., https://www.storemaven.com/academy/how-to-choose-an-app-category/ ("The category you choose needs to be relevant to your app, of course. But it should also enable your app to rank well in category charts. Users often discover new apps to download by perusing the charts. This means that high-ranking apps generally receive a lot of traffic and are able to generate more organic downloads… The category you choose for your app can affect its discoverability and conversion rates in both the Apple App and Google Play stores. The trick to choosing the right category for your unique app is balancing relevance with your competition."); https://support.google.com/googleplay/android-developer/answer/9859673 ("You can choose a category and add tags to your app or game in Play Console. Categories and tags help users to search for and discover the most relevant apps in the Play Store.").

250 See Appendix D.

251 See GOOG-PLAY-003335786.R at 808.R (Google has considered "charge for Play services individually"). [redacted]

226. In the actual world, Google made many investments focused on developers who generate revenue from consumer spend, such as subscription services, numerous different forms of payment, and others. In Developer Plaintiffs' but-for world, if Google were generating revenue in some other way, those investments would likely have been refocused and the putative developer class members that generate revenue from consumer spend may not have had the same tools and services in the but-for world from Google. Whether developers could have obtained the same tools at the same cost depends on the tools needed by the developer and their cost from other sources. Developers that earn revenue from IAPs could have been worse off than developers that earn revenue from ads, for example, because Google would have had economic incentives to invest more in tools for ads than in tools for IAPs.[252]

227. Certain payment processing services are more costly than others and if in the but-for world Google were forced to reduce its service fee rate, some of those services may not have been provided. For example, as discussed above, approximately [redacted] of the U.S. consumers use DCB.[253] The cost to Google of processing DCB is [redacted] on average, and the estimated cost of processing DCB to developers is [redacted] – as much as [redacted].[254] If some developers have relatively more DCB payments, the cost to Google to provide services to these developers would be relatively high, making it unlikely that Google would make a uniform reduction to the service fee rate to 15%. Either Google would have service fee rates that vary across the forms of payment or Google would not offer those payment forms that are particularly costly.[255] Developers that rely more on DCB would have been worse off than developers not reliant on that payment form. Individualized inquiry is required to determine which developers would be better off and which developers would have been worse off in the but-for world.

228. If, in the but-for world a developer had to spend more to obtain services than it saved in lower service fees, the developer would have been worse off. About [redacted] of the putative developer class members generated $100 or less in consumer spend across all years of the class period and therefore would have had (at most) [redacted] in lower service fees in the but-for world.[256] If, in the but-for world, any of these developers had to separately purchase a service that was previously obtained through Google Play, and the cost of that service was greater than $15, the developer would have been worse off in the but-for world.

[redacted]

252 Marchak Ex. 375 (GOOG-PLAY-007628059) at 065 ("Because of a choice 12 years ago to charge 30% for premium downloads, we've invested in IAP, FOP coverage, DCB, Gift Cards, Growth Consulting, Subs, Points. Imagine if we decided to charge 30% of all ad sales instead?").

253 See Exhibit 38.

254 See GOOG-PLAY-000337564 at-587.

255 See GOOG-PLAY-006990552 at 555.

256 Exhibit 9 and Developer Complaint at ¶244.

VII. DEVELOPER PLAINTIFFS' EXPERTS HAVE NO BASIS FOR THEIR CLAIM THAT ALL DEVELOPERS WOULD HAVE BEEN SUBJECT TO A LOWER SERVICE FEE RATE IN THE BUT-FOR WORLD

229. Developer Plaintiffs' expert, Dr. Sibley, offers an overly simplistic description of the but-for world and opinion of common impact. According to Dr. Sibley, absent the alleged conduct, "more alternative app distribution options would be available to a larger number of developers of Android apps," and the availability of those "competitive options" would have the effect of aligning prices more closely to costs.[257] Dr. Sibley states that in the but-for world, Google would face competition for all developers because all developers would "enjoy competitive alternatives," and faced with such competition, "it is likely that Google would reduce its default service fee offered to all developers below 30%"[258]

230. In this section, I discuss Dr. Sibley's opinion on classwide impact. First, I show that Dr. Sibley's finding of classwide impact is contradicted by his own finding that some developers "resisted" Google's alleged conduct and negotiated "competitive" service fee rates in the actual world.[259] Dr. Sibley's admission that some but not all developers obtained "competitive" rates in the actual world shows that he ignores differences across developers that would continue to exist in the but-for world so that only some, but not all, developers in the but-for world would obtain lower service fees.

231. I next address the bases for Dr. Sibley's conclusion that all developers were impacted because in the but-for world, all developers' apps supposedly would benefit from increased competition and, as a result, all developers' service fee rates would have been lower.[260] Dr. Sibley's opinion is an assumption. Even accepting Dr. Sibley's claim that there would have been an increase in competition in the but-for world, it is not necessarily the case that all developers would obtain lower service fees from the increase in competition. Classwide impact cannot be assumed. An analysis is required to determine whether each developer would benefit from such an increase in competition.[261] Determining whether a developer

---

257 Sibley Report at ¶211, 250.

258 Sibley Report at ¶258.

259 Sibley Report at ¶¶ 255-57. Dr. Williams, another Developer Plaintiff Expert, utilizes the service fee rates obtained by these putative developer class members as a benchmark for the but-for service fee rate. See Williams Report at ¶60.

260 Dr. Sibley claims that all developers would have lower service fees in the but-for world. See Sibley Report at ¶258 ("In the but-for world, Google would face competition for *all* developers, including those paying the 30% service fee, because *all* developers would enjoy competitive alternatives.") [emphasis in original]

261 In other industries, increases in competition have been found to benefit some, but not all customers. For example, increased competition on airline routes have been found to reduce prices for leisure travelers but not change prices for business travelers. See Stavins, Joanna, "Price discrimination in the airline market: The effect of market concentration," *The Review of Economics and Statistics*, Vol. 83, No. 1, 2001, pp. 200-202. Similarly, researchers found that in analgesics, competition leads to lower prices on Tylenol's price, while increasing the store-brand's price. See Chintagunta, Pradeep K., "Investigating category pricing behavior at a retail chain," *Journal of Marketing Research*, Vol. 39, 2002, pp. 141-154.

would have been better off in the but-for world depends, at least, on that developer's individual characteristics, which Dr. Sibley failed to analyze.

A. Developer Plaintiffs' Experts Concede that Certain Developers were Not Impacted Because They Were Able To Obtain the Alleged "Competitive" Service Fee rate

232. According to Dr. Sibley, there are numerous developers that were able to "resist" Google's alleged conduct and obtain "competitive" service fee rates, and thus were not impacted.[262]

233. Dr. Sibley acknowledges that in the actual world developers are different. He admits that some developers that sought alternative billing or distribution options and/or had relatively higher value to Google Play were able to negotiate lower service fee rates or enhanced services.[263] According to Dr. Sibley, these are "examples [that] illustrate how Google responds to competition."[264] Thus, Dr. Sibley recognizes that in the actual world, Google responded to competition by providing some – but not all – developers with lower rates and more services and those developers obtained competitive service fee rates.

234. Dr. Sibley identifies numerous developers that, in his words, were able to "resist" the alleged conduct. These developers include those that obtained lower service fee rates as part of programs like LRAP and ADAP, the developers that obtained lower rates in 2018 when Google lowered the rate for subscription IAPs, and the developers that negotiated special deals with Google as part of Project Hug and Games Velocity Program.

235. Dr. Williams, another expert for Developer Plaintiffs, also asserts that because Google faced "meaningful competition," it had to reduce its service fee rates to 15% or lower for certain developers.[265] Dr. Williams, like Dr. Sibley, identifies the developers in Google Play's LRAP, LRAP++, ADAP, and ADAP++ programs. According to Dr. Williams, a 15% service fee rate is a competitive benchmark for a service fee in the but-for world.[266]

---

262 Sibley Report at ¶¶102-103 ("In limited circumstances where Google has been forced to compete – particularly with respect to developers who have resisted Google's tie and used IAP solutions other than Google Play Billing – Google has sharply reduced its service fees."). Dr. Sibley identifies developers that participate in programs such as LRAP, ADAP, and LRAP++, and developers with subscription IAPs that were renewed after one year. See also Sibley Report at ¶¶182, 188, 191-193, 255-256; Williams Report at ¶¶50-60.

263 Sibley Report at ¶239 describing Google's Project Hug ("Google identified [redacted] major game developers that drove 'disproportionate value to Google' (defined as approximately [redacted] of total Play consumer spend), were in some danger of leaving Play for alternative distribution routes and had the capability to '"go-it-alone" on Android.'").

264 Sibley Report at ¶257.

265 Williams Report at ¶60; see also Williams Report at ¶¶50-59.

266 Williams Report at ¶63. According to Dr. Williams, the use of Google Play's service fee rate for developers that were able to "resist the tie," is a "strong yardstick" for a competitive service fee rate. See Williams Report at ¶67. According to Dr. Sibley, there were [redacted] developers that were able to "resist" the alleged tie. See Sibley Report at ¶188.

236. Dr. Sibley provides no reason (or evidence) that these developers would obtain further reductions in service fee rates in the but-for world and does not explain why the developers could have been impacted on transactions that occurred at other rates or in time periods prior to Google Play's programs, given that those developers were able to "resist" Google Play's alleged conduct. Moreover, Dr. Williams' use of these developers as a competitive benchmark contradicts that there would have been any further rate reductions for those developers in the but-for world.[267] Those developers that were able to negotiate a 15% rate, or lower, then, were not impacted by the alleged conduct.

B. Developer Plaintiffs' Expert's Opinions Regarding Common Impact Are Not Supported By Economic Theory or the Evidence

237. Dr. Sibley offers five arguments to support his opinion that all developers would have lower service fees in the but-for world. However, these arguments are not supported by evidence of the actual world or by economic logic.[268] I discuss each below.

1. Dr. Sibley's Claim that Competition Would Have Been More Robust for All Developers Ignore Facts of the Case

238. Dr. Sibley's argument that all developers' service fee rates would have been lower in the but-for world because "competition for all developers would be more robust" ignores what he has acknowledged about the actual world – that there are meaningful differences across developers and how competition impacts developers.[269] Dr. Sibley admits that differences across developers led to lower service fee rates for some developers, but not all developers. Developers that have been able to negotiate particular deals with Google Play are the developers with popular apps that can more easily attract and divert users to competing platforms.[270] As I explained above, this is true not just for Google Play, but for app stores

---

267 Williams Report at ¶67 ("The material distinction between the yardstick market and the broader Android IAP platform services is the effect of the challenged conduct because Google had not enforced its tie as to the yardstick developers.").

268 Sibley Report at ¶¶259-275. It is my understanding, based on Dr. Williams' report, that he is offering expert opinions related to damages, not classwide impact. See Williams Report at ¶61 ("In the but-for world, as Dr. Sibley shows in his report, all developers would have the option of distributing their apps and in-app products…and incentivize Google to reduce its service fee across the board."). Moreover, Dr. Williams does not study or analyze the question of classwide impact, with his damage analysis instead being predicated on the assumption that all developers are the same and the but-for service fee rate is the same for all developers.

269 Sibley Report at ¶¶259-260.

270 Dr. Sibley acknowledges that "network effects" are relevant to app store competition such that rival platforms compete by attempting to obtain a "critical mass of users and developers." See Sibley Report at ¶116. Obtaining such a "critical mass" most efficiently involves attempting to persuade developers with the most popular apps to switch stores or to stay with a store and likewise persuade app users that spend the most to switch stores or to stay with a store. For rival platforms that wish to generate revenue from sales of apps and IAPs, it makes economic sense to concentrate on getting the most popular revenue-generating apps and the highest value app users (i.e., users with the greatest amount of spend) to switch. As shown above (see Exhibits 14.5 and 19) revenues and spend are concentrated among a small

more generally; that is, app stores generally compete not by offering service fee rate reductions to all developers, but instead by offering service fee reductions to large developers with popular apps that can drive users to their stores.[271]

239. The ability to drive consumers and revenue from one store to another, and thus, the ability to negotiate a lower service fee rate (or expanded services), is different across developers, or at least across different types of developers. This explains the rate reductions observed in the actual world and demonstrates that service fee rate reductions would not occur for all developers in the but-for world.

240. In the but-for world, the assumed increase in competition would therefore be expected to lead to differential effects across developers – those developers that can drive consumers from one platform to another through the popularity of their apps will be the developers that are more likely to obtain lower service fees or additional services.

241. Furthermore, Dr. Sibley's argument that competition for developers by alternative payment processors "would be expected to provide all class members with competitive options for IAP transaction services in the but-for world" does not consider that for a substantial percentage of developers, Google Play's service fee rate likely is already the most competitive option. As I described above, Google Play's service fees are lower than the fees some developers would need to pay if they separately contracted with payment processors.[272] Even setting aside the other valuable services Google Play provides in addition to payment processing, these other processors' rates would be higher than 30% for certain developers. For those developers that offer apps or IAPs at low price-points, independently contracting with payment processors is not an economic option. 22% of putative developer class members offer apps, subscriptions, and IAPs at prices of $0.99 (only) and therefore the payment processing costs of many processors for those developers may be higher than 30%.[273] 36% of developers offer apps, subscriptions, and IAPs at prices of $1.99 or lower. Other developers that offer apps and IAPs at multiple different price points, including these low price-points, could also find that separately contracting with payment processors is not more affordable, depending on the mix of their transactions. Identifying those developers would require analyzing the product-price mix (the percentage of transactions at each price point) and determining whether the proportion of low-price point sales causes the 30% rate to be higher or lower than the rate a developer would have if it independently contracted with a payment processor. Contrary to Dr. Sibley's assumption, not all developers necessarily would have had the option of lower service fees in the but-for world, and therefore it cannot be assumed that they are impacted by the alleged conduct.

---

percentage of developers and app users. Therefore, one would not expect competition to be concentrated on all app developers or all app users.

271 GOOG-PLAY-000568027 at 028; See EPIC_GOOGLE_01413875 at 877 ("[redacted]).

272 See Sibley Report at ¶¶261, 198.

273 See Exhibit 31.

Determining whether contracting separately with a payment processor is an economic option requires an individualized analysis of the developer's apps, prices, and transactions.

2. Dr. Sibley's Claim that All Developers in the But-For World Have a Better "Best Option" Is Without Basis

242. Dr. Sibley's assumption that in the but-for world, a developer would have been able to negotiate lower service fee rates because the developer's "best option if negotiations break down" would have been better is not true for all putative developer class members.[274]

243. Dr. Sibley's assumption that a developer's "best option" would have been better in the but-for world may not be true for at least 22% of putative developer class members that rely solely on prices of $0.99 and would not have had a better option in the but-for world for payment processing.[275] Thus, it is likely that at least 22% of the putative developer class members cannot show antitrust impact. Moreover, the percentage could be higher because more developers, depending on their mix of transactions across different price points also would not have better options than a 30% service fee rate. This includes the 36% of developers that price apps, subscriptions or IAPs at $1.99 or lower.[276] Individualized inquiry of each developer and their transactions is thus required to determine whether any developer is better off or worse off in the but-for world and whether the developer can show antitrust impact.

244. Additionally, developers that rely on relatively more costly forms of payment (e.g., DCB and Play gift cards) similarly would have been at a relative disadvantage compared to other developers in the but-for world. The costs of providing those services are higher than the cost of payment processing of other forms of payment.[277] For some putative developer class members, DCB sales account for a substantial amount of overall sales. For [redacted] 97% of sales for [redacted] and all of [redacted] sales.[278] Dr. Sibley hasn't considered whether developers that rely on those forms of payment would have a better "best option" in the but-for world.

245. Dr. Sibley fails to acknowledge that whether developers' "best option" would have been better in the but-for world generally depends on the nature of the purported increased competition in the but-for world. An increase in competition from an expansion of the Samsung Galaxy Store would not present a better "best option" to developers that rely on consumers with mobile devices other than Samsung (since that app store serves only Samsung's devices).[279] Entry by a company like Epic, a game app developer, may provide

274 Sibley Report at ¶¶263-264.

275 See Exhibit 31.

276 See Exhibit 31.

277 See e.g., GOOG-PLAY-000437819 at 838 (showing transaction costs for DCB of [redacted] to [redacted] in 2016) and GOOG-PLAY-000337564 at 587 (showing DCB costs to Google of [redacted] and cost for developer to replicate of [redacted]; and showing gift card costs to Google of [redacted] and cost for developer to replicate of [redacted]).

278 GOOG-PLAY-005535885 and GOOG-PLAY-010801689.

279 Deposition of Lawrence Koh, December 9, 2021, ("Koh Dep.") at p.323.

an additional option for game apps but may not for non-game apps.[280] And while the Amazon Appstore serves Android devices, it has been Amazon's competitive strategy to compete aggressively for certain gaming developers with popular apps and to focus on the consumers that make substantial purchases – typically those that use gaming apps.[281] So, expansion by Amazon or entry by an app store like Amazon would not necessarily provide a better option to all developers.

246. Finally, even if a "better option" exists for a developer in the but-for world, the developer may not have the ability to negotiate a lower rate if it does not have a popular, revenue-driving app. Such developers have less negotiating leverage because even if the developer left Google Play (or any other app store) it would not drive consumers away from Google Play and toward some rival store. Moreover, according to Dr. Williams, developers generally do not pass through savings on reduced service fee rates to consumers and so, according to his opinion, developers would not use differences in retail prices to drive consumers from one store to another.[282]

3. Dr. Sibley's Claim Regarding Competition in PC App Stores Contradicts His Conclusion of Classwide Impact

247. Dr. Sibley's claim that all developers would have lower service fee rates because Google would respond to competition like Microsoft responded to the entry of Epic in PC app distribution is based on a mistaken and incomplete understanding of app stores' response to Epic's entry.[283]

248. The Epic Game Store distributes game apps for PCs; it began operations in December 2018 and charged a 12% service fee rate.[284] The two primary PC game app stores at the time of Epic's entry were (and continue to be) Microsoft and Steam. Microsoft distributes game apps for its Xbox gaming consoles and both game apps and non-game apps for PCs with

---

280 See Exhibit 11 showing [redacted] of "U.S." developers offer only non-game apps during the class period.

281 See GOOG-PLAY-000310705. See also Feng Dep. Ex. PX 531 at p. 19 ("[Amazon is] focused on attracting titles with high % of HVU spend and migrating the HVUs to Amazon apk"); Feng Dep. Ex. PX 532 at p. 4 ("Competitive Android stores such as Amazon Appstore in Japan have reached 2-10% of developer revenue, mainly by attracting Play HVUs with heavy discounts").

282 Williams Report at ¶73. As discussed below, whether developers pass through reduced service fees requires an individualized analysis of a developer's apps and the economic factors that affect pass-through.

283 Sibley Report at ¶¶265-267.

284 https://techcrunch.com/2018/12/06/epic-games-store/ (article dated December 7, 2018, reporting that the Epic Games Store "quietly went live today"); https://www.epicgames.com/site/en-US/epic-games-store-faq ("The Epic Games Store currently offers PC and Mac support. You can check platform compatibility for individual titles by referring to the "About Game" section of any product page." "…creators will earn 88% of all the revenue from their game…").

Windows operating systems.[285] Prior to Epic's entry, Microsoft charged a service fee rate of 30% for game apps on PCs, the same types of apps as those distributed by Epic, and a 30% rate for non-game apps. Microsoft maintained its 30% service fee rate for game apps on PCs until August 2021 – over two years after Epic's entry.[286] It is unclear whether the entry of Epic was even a factor that led Microsoft to reduce its rate on game apps when it did. Even if Microsoft's rate reduction was due to increased competition from Epic, PC game app developers would not have been impacted by the increased competition for more than two years. If competition in mobile game apps is like competition in PC apps, as Dr. Sibley's comparison implies, many developers would not be impacted. For example, 32 months elapsed between the Epic Store's entry and Microsoft's rate reduction. If a new app store entered at the onset of the class period with a lower service fee rate and Google (and others) responded with rate reductions 32 months later, 38% of putative developer class members would not benefit from that entry because over those 32 months, those putative developer class members stopped posting sales.[287]

249. The service fee rates of Steam, the other primary PC game app distributor, do not support Dr. Sibley's opinion that increased competition would lead to lower rates for all developers. In November 2018 (prior to Epic's entry), Steam reduced its service fee rate from 30% to 25% for game apps with consumer spend of $10 million to $50 million and to 20% for game apps with consumer spend over $50 million.[288] It kept its service fee rate at 30% for game apps with consumer spend of less than $10 million and has continued to charge 30% for those game apps.[289]

250. Microsoft and Steam's reductions of fees for certain developers and not others contradict Dr. Sibley's claim that increased competition in the but-for world would lead to lower rates for all developers.[290] Those app stores' reductions are, instead, consistent with developers

---

285 https://www.xbox.com/en-US/microsoft-store ("A Store that is twice as fast, easier to navigate, and safer for the whole family to find your next favorite games and entertainment on your Xbox."); https://www.microsoft.com/en-us/store/apps/windows

286 Booty, Matt, "Continuing Our PC Gaming Journey 2021 and Beyond," Xbox, April 29, 2021, https://news.xbox.com/en-us/2021/04/29/continuing-our-pc-gaming-journey-in-2021-and-beyond/ ("starting on August 1 the developer share of Microsoft Store PC games sales net revenue will increase to 88%, from 70%"); https://www.protocol.com/bulletins/microsoft-store-commission-cut. Microsoft announced its rate reduction in April 2021.

287 See Exhibit 43.

288 Steamworks Development, Steam Community, November 30, 2018, https://steamcommunity.com/groups/steamworks/announcements/detail/1697191267930157838, accessed November 8, 2021.

289 See Exhibit 28; https://steamcommunity.com/groups/steamworks/announcements/detail/1697191267930157838, accessed November 8, 2021.

290 Additionally, after ONE Store in Korea reduced its rates from 30% to 20% (and to 5% to developers that use their own payment processing), there was no corresponding across-the-board

having varying importance to app stores and app store competition and consistent with my opinion that service fee rates may be lower in the but-for world for some, but not all, developers.

4. Dr. Sibley's Claim that Google Documents Support the Conclusion that All Developers Would Have a Lower Service Fee Is Not Accurate

251. Dr. Sibley's argument that Google's business documents indicate a 30% rate [redacted] does not imply that all developers would have lower rates in the but-for world.[291]

252. Many Google documents show that Google considered changes to the service fee rate that would have meant that some, but not all, developers' rates would have been lower. Google documents show it considered a strategy of [redacted][292] Google also considered rates that could be different for [redacted].[293] Google documents show that it considered service fee rate reductions [redacted][294] That strategy implies variation in rates across developers with some developers' rates remaining at 30%.[295] There are also Google documents that show that Google considered "custom deals" to particular developers like Project Hug and its Accelerator Programs.[296]

5. Dr. Sibley's Claim that Google Would Not "Price Discriminate" and Set Different Service Fee Rates for Different Developers Is Inaccurate

253. Finally, Dr. Sibley's claim that Google would not "price discriminate" and charge developers different service fee rates because "a programmatic reduction of the 30% default rate would be most consistent with Google's economic incentives and conform to Google's historic pricing practices"[297] is not true.

---

reduction in rates by other app stores in Korea, including Google Play, further contradicting Dr. Sibley's view of the but-for world. Instead, Google Play targeted particular developers through its Project Hug program. See, for example, GOOG-PLAY-000005203- 08; GOOG-PLAY-003332070-081. See also, Exhibit 28 and see this report's production; "One Store Introduction," One Store Corp., January 2020, https://dev.onestore.co.kr/devpoc/static-res/ files/ONEstoreIntro_dev_en.pdf, accessed November 8, 2021, at slide 9.

291 Sibley Report at ¶268.

292 See e.g., GOOG-PLAY-000444214.R; GOOG-PLAY-006990552 at 554; GOOG-PLAY-003335786.R at 808.R.

293 GOOG-PLAY-003331592.R.

294 GOOG-PLAY-003331592.R at 606.R; GOOG-PLAY-006990552 at 554.

295 GOOG-PLAY-000565001 (August 2019).

296 GOOG-PLAY-003331592.R .

297 Sibley Report at ¶270. Certain Google Play rate reductions could be considered "programmatic," in that they were targeted to certain types of developers. For example, LRAP targeted certain types of

254. Google's "historic pricing practices" do not include a uniform reduction to all developers like the reduction claimed by Dr. Sibley (and Dr. Williams). Google's service fee reductions have been offered to certain developers (e.g., the LRAP developers), developers with certain amounts of consumer spend (e.g., consumer spend of $1 million or less), or certain kinds of apps (e.g., apps with subscription IAPs).[298] Moreover, some developers individually negotiate service fee rates and service terms with Google – which clearly do not reflect the type of reduction claimed by Developer Plaintiff experts.[299]

255. Dr. Sibley has not provided any explanation or evidence that competition in the but-for world would have been different.

256. Dr. Sibley's claim about charging different prices to different developers or "price discrimination" is also contradicted by the facts. He states that this is rare and "normally done only for a small number of large customers, whose business is worth the substantial transaction costs required to price them both individually and profitably."[300] However, Google has provided lower rates to large customers whose businesses were apparently worth whatever transactions costs were required. In addition, over the class period, Google has continued to increase the number of developers to which it offers lower service fee rates. In 2017, [redacted] developers had obtained rates lower than 30%. In 2020, there were [redacted] developers that had obtained rates lower than 30%.[301]

257. Dr. Sibley's opinion that Google would have been unable to set different rates to different developers because doing so "requires a level of detailed knowledge about individual tastes that firms typically do not possess" does not consider recent developments in technology and in the economic literature.[302] Widespread use of computers and big data in the past twenty

developers. Dr. Sibley's use of the terms "programmatic," in that sense would describe Google Play rate reductions but would not be consistent with his opinion that Google Play would reduce rates to *all* developers.

298 For example, in January 2018, Google reduced the subscription service fee to 15% for the second year of subscriptions. "Playtime 2017: Find success on Google Play and grow your business with new Play Console features," Android Developers Blog, October 19, 2017, https://android-developers.googleblog.com/2017/10/playtime-2017-find-success-on-google.html (under "Grow your subscriptions business" header). In September 2020, Google stated a policy "clarification" around a digital content exception. In March 2021, Google announced a 15% service fee on the developer's first $1 million, effective July 1, 2021. "Boosting developer success on Google Play," Android Developers Blog, March 16, 2021, https://android-developers.googleblog.com/2021/03/boosting-dev-success.html. In October 2021, Google announced a 15% service fees on subscriptions from the outset (i.e., including first year). "Evolving our business model to address developer needs," Android Developers Blog, October 21, 2021, https://android-developers.googleblog.com/2021/10/evolving-business-model.html.

299 See Deposition of Mike Marchak, January 12-13, 2022, ("Marchak Dep."), Vol. 1 at pp. 87-88, 309-310.

300 Sibley Report at ¶274.

301 See Exhibit 26.

302 Sibley Report at ¶274. Moreover, Dr. Sibley's opinion regarding "price discrimination" misunderstands the issue of classwide impact. The question of classwide impact is whether all developers

years has resulted in "price discrimination" not only being possible but used by many companies. Dynamic pricing, personalized pricing, and algorithmic price discrimination, all result in different prices to different customers, and are a commercial reality.[303] Moreover, Google Play has been able to charge different service fee rates to developers based on the length of time an individual consumer has subscribed to the developer's app, further undermining Dr. Sibley's opinion.

C. Developer Plaintiffs' Expert's Opinion Related to Lost Profits Is Flawed Because It Ignores Many Individualized Factors

258. Determining whether a putative developer class member is impacted, and measuring its damages, should be based on a comparison of the developers' economic situation in the actual and but-for worlds.[304] In this case, a developer's economic situation cannot be limited to its service fee rate because determining and measuring any losses to the developer from the alleged conduct implicates more than the service fee rate. If, in the but-for world, the developer passed through some portion of lower service fees, then an analysis of the developers' revenues must be undertaken; similarly, if the developer incurred higher costs in the but-for world, then an analysis of those costs must be performed. These considerations mean that the developer's profits in the actual and but-for worlds must be compared.

259. Comparing profits in the actual and but-for worlds is more complicated than comparing service fee rates. Determining profits in the but-for world requires an analysis of the developer's revenues as well as its costs. Its revenues will be affected by the price elasticity of demand for the developer's apps. Its costs will depend on the characteristics of the apps, including, for example, the technical complexity of the developer's apps and the costs

---

would get a lower rate – for that conclusion to fail it is not necessary that every developer gets a different rate.

303 Airlines, Uber, Amazon, and Staples have all used complex pricing methods. For dynamic pricing literature, see, for example, McAfee, R. Preston and Vera te Velde, "Dynamic Price Discrimination in the Airline Industry," *Handbook on Economics and Information Systems*, in T.J. Hendershott, *Handbook on Economics and Information Systems*, Vol. 1, Elsevier, 2006; Aguirregabiria, Victor, Allan Collard-Wexler, and Stephen P. Ryan, "Dynamic Games in Empirical Industrial Organization," in *Handbook of Industrial Organization*, Vol. 4, No. 1, 2021, pp. 225-343; "Big Data and Personalized Pricing," Council of Economic Advisors Report, 2015, https://obamawhitehouse.archives.gov/sites/default/files/whitehouse_files/docs/Big_Data_Report_Nonembargo_v2.pdf and a non-technical overview here: https://obamawhitehouse.archives.gov/blog/2015/02/06/economics-big-data-and-differential-pricing. See also Shiller, Benjamin Reed, "First-Degree Price Discrimination Using Big Data," Working Paper No. 58, Brandeis University, Department of Economics and International Business School, 2014, https://www.brandeis.edu/economics/RePEc/brd/doc/Brandeis_WP58R2.pdf. For algorithmic price discrimination, see, for example, Bar-Gill, Oren, "Algorithmic Price Discrimination When Demand Is a Function of Both Preferences and (Mis)perceptions," *University of Chicago Law Review*, Vol. 86, No. 2, 2019, pp. 217-254.

304 See Williams Report at ¶15 citing Reference Guide on Estimation of Economic Damages ("Damages measurement then determines the plaintiff's hypothetical value in the but-for scenario. Economic damages are the difference between that [but-for] value and the actual value that the plaintiff achieved.")

associated with offering and updating the apps in different app stores, as well as other factors. Because of the variation across developers and the factors that affect developers' profits, determining impact and calculating damages based on a lost profits analysis requires individualized information and analysis.

260. While Dr. Williams apparently agrees that determination of impact and damages is based on a consideration of a plaintiff's economic situation in the actual world and the but-for world, he attempts to avoid the more complex and individualized analysis of putative developers' profits.[305] He claims that his calculated developer "overcharge" – that is, the difference between the actual and but-for service fees – understates all developers' lost profits.[306] Dr. Williams apparently concludes that analyzing whether developers lost profits in the but-for world is not necessary. However, Dr. Williams fails to consider several important factors that would affect a comparison of developer profits in the actual and but-for world.[307]

1. Developers' Prices in the But-For World Would Have Been Affected By Competition Across Developers

261. Dr. Williams' claim that analyzing developers' lost profits is not necessary because developers would not change retail prices in the but-for world does not consider economic realities.[308]

---

305 Williams Report at ¶¶15, 71.

306 Williams Report at ¶71 ("If a developer were to reduce its app prices in response to lower service fees in the but-for world (i.e., pass through those lower fees to its customers), basic economics predicts that the developer likely would have sold more apps at the lower price and would have generated additional profits in doing so.").

307 While Dr. Williams states that damage methodologies, generally and including his, should be based on a comparison of the plaintiff's position in the actual world and the but-for world, where only the effects of the alleged conduct are allowed to be different between those two worlds, he fails to recognize that principle is not a legitimate basis to ignore that in the but-for world some developers' costs would have been higher or that Google would change the way it monetizes Google Play. Williams Report at ¶¶70-71. That is, Developer Plaintiffs claim that absent the alleged conduct there would have been more app stores. If a developer's costs rise because the developer is now forced to serve more app stores, those costs must be considered as part of the but-for world. Similarly, if in the but-for world Google Play's revenues from a service fee assessed as a percentage of consumer spend declines substantially, and if it can change the way it collects revenue from the app store and (as is the case) has business documents that record and analyze those alternative ways to collect revenue, then not only does Google have the clear incentive to do so in the but-for world, but taking account of those changes is proper and necessary in an analysis of impact and damages.

308 Dr. Williams states that a developer that passed through some (or all) of lower service fees would generate "additional" profits from doing so. Williams Report at ¶71. However, Dr. Williams' formula, set out in Appendix III, does not show that a developer would necessarily earn more profits if it passed through some portion of any lower service fees. The formula offered by Dr. Williams shows that lost profits are greater than or equal to some portion of the overcharge, not considering any change in sales due to lower prices. So, for example, if the pass-through rate is one (100% pass-through), Dr. Williams' formula states that the developer's lost profits are greater than *or equal to zero*, not that they are greater

262. Dr. Williams does not consider competition across developers or the way in which prices are set. A profit-maximizing developer that faces competition and sets prices to end in "99," for example, may not find a price reduction from one price point to another to be profit-maximizing and therefore would choose not to reduce its price when service fee rates fall.[309] However, if its rival does reduce prices, that app developer may lose profits due to competition. The "calculus" associated with such pricing actions and reactions on the part of developers is complicated, but without analyzing them, Dr. Williams' conclusion that developers' profits will always be higher in the but-for world is an assumption, not the result of any economic analysis.

2. Developers' Costs Would Have Been Different in the But-For World

263. Dr. Williams assumes that developers' costs are the same in the actual and but-for worlds.[310] Dr. Williams provides no basis for this assumption and the evidence shows that it would not be true for all developers. A lost profits analysis that addresses differences in cost between the actual and but-for worlds requires a detailed analysis of individual developers' costs and profits.

264. In Dr. Sibley's but-for world, some developers' costs would have been higher. Dr. Sibley claims that the alleged conduct includes certain contracts between Google and mobile device OEMs that relate to the fragmentation of the Android operating system.[311] Therefore, absent those allegedly anticompetitive contracts, there would be more stores in the but-for world, and those stores would have run on more incompatible versions of Android. Given the differences across variants of Android and app stores, a developer listing an app in multiple stores would have been forced to create different versions of the app (and different versions of each update) which implies higher distribution and development costs for some developers. For example, when Epic was developing Fortnite for Android, Epic had to invest engineering time and resources to determine whether Fortnite would operate on different versions of Android. Developing Fortnite for incompatible Android OSs would have taken even more effort and resources.[312]

---

than or equal to the overcharge. Moreover, the formula does not consider any change in profits from a change in sales – that amount is explicitly set to zero in Dr. Williams' formulation, which according to Dr. Williams, is a conservative approach because he assumes a positive change in sales in the but-for world. This is a flawed assumption, because some apps may lose sales even if it sets a lower price, because of more intense price competition that may occur in the but-for world.

309 Exhibit 32 shows that 97% of app download and IAP prices end in "99."

310 See Williams Report Appendix III, ¶109 ("I assume that unit cost for developers is C and is fixed between actual and but-for worlds.").

311 See e.g., Sibley Report at ¶222 ("AFAs [Anti-Fragmentation Agreements] and ACCs [Android Compatibility Commitments] restrict OEMs' ability to manufacture and sell any device that uses a version of Android that is not approved by Google.").

312 See [redacted] More generally, developers may incur costs to distribute apps in different app stores. See also Koh Dep. at pp. 320-321 ("even working with a one-stop shop [that would take our code then make it available for these different app stores] that there was engineering work required for us to be able to utilize those services.").

265. Developers' costs in the but-for world also may be higher for other reasons. There are costs associated with transacting with each app store, including, for example, the costs in understanding and complying with each app store's terms and conditions, updating an app in each store, and marketing the app in each store. As a developer offers an app in more stores, those costs can increase.[313]

266. Given that developers' costs could increase in the but-for world, determining whether a developer has lost profits as compared to profits in that but-for world must consider those costs. It cannot be assumed, as Dr. Williams has assumed, that costs are the same. Determining whether a developer's costs are higher, however, requires an individualized analysis of the developer's apps.

3. Google Play Benefits Would Have Been Different in the But-For World

267. Moreover, services and tools provided by Google Play in the actual world may not be available from all app stores in the but-for world or available on the same terms as Google Play provides them. Different developers value services and tools differently.[314] For example, not all app stores (or alternative payment processors) may provide the more costly forms of payment such as DCB[315] and gift cards, and developers whose customers rely more on those forms of payment would either experience higher costs or lose sales.[316]

268. Google Play makes numerous other services and tools available to developers. Whether each of those services are important to a developer and would have been available in the but-for world at the same cost as in the actual world requires a developer-specific analysis.[317]

4. Google Play's Service Fees Could Be Assessed Differently in the But-For World

269. Dr. Williams' opinion regarding lost profits also does not properly account for Google's actions in the but-for world.[318] That is, Dr. Williams' opinion does not allow for the

---

313 See [redacted]

314 GOOG-PLAY-0110236932.

315 As described above, some putative developer class members rely on DCB sales. For [redacted] 97% of sales for [redacted] and all of [redacted] sales. See this report's production.

316 GOOG-PLAY-000337564 at -587; GOOG-PLAY-000565541 at -559.

317 Glick Dep. at p. 214.

318 Dr. Williams describes that in a damages analysis, a "characterization of the harmful event…will include a description of the defendant's proper actions in place of its unlawful actions and a statement

possibility that Google would change the way it monetizes Google Play in the but-for world even though, according to Developer Plaintiffs, Google would not earn any revenue from developers that used another billing system instead of Google Play Billing. Service fees generated from subscriptions and IAPs accounted for [redacted] of total Google Play revenue associated with "U.S." developers during the class period.[319] Nearly all of Google Play's revenues would have been "at-risk" in Developer Plaintiffs' conceived but-for world. Faced with such risk, it is more than economically plausible that Google would respond in some way.[320] Therefore, the way Google would respond must be considered in any determination of developer impact and calculation of developer damages. Calculating a developer's lost profits would require an individualized analysis of how an alternative monetization strategy would affect the developer.[321]

270. There are several ways Google could change Google Play's monetization to address the potential loss of revenues from developers that chose to use alternative payment processors in the but-for world. For example, Google could charge [redacted].[322] For some developers, this annual fee would mean higher costs. For example, Peekya, one of the developer class representatives generated [redacted] in consumer spend during the 25-month period that it offered apps in Google Play. Its service fees were [redacted]. Peekya also paid a $25 registration fee. So, in total, Peekya's fees to Google Play were [redacted].[323] If Google Play had charged [redacted], Peekya would have paid [redacted] over the 25 months.[324] Total fees payable to Google Play would have been

---

about the economic situation absent the wrongdoing, with the defendant's proper actions replacing the unlawful ones (the but-for scenario)." See Williams Report at ¶15.

319 See Exhibit 4.

320 Feng Dep. at pp. 158-159 ("If users are led to a way to pay outside of Play billing today, we would not be able to collect a service fee. I think if that was the main mechanism that purchases are made through Play-distributed apps, it's likely we would find mechanisms to charge a service fee.").

321 Developer Plaintiffs apparently contend that the only way that Google can respond to competition is through a reduction in the service fee rate. But Google has the ability and economic incentive to pursue other ways to respond to competition if those other strategies generate more revenue and profits than a reduction in the service fee rate. In addition, other monetization strategies may lead to a higher quality app store and would have been consistent with Google finding other ways to compete in the but-for world, besides just service fee rates. See for example, GOOG-PLAY-000565001.R at 013 and 019 (describing that [redacted] would discourage spam and low-quality apps). Moreover, as shown here, an alternative monetization strategy is not purely hypothetical. Google has considered and analyzed alternative strategies for Google Play.

322 See GOOG-PLAY-006990552 and GOOG-PLAY-000565001.R. Other amounts considered were [redacted] Google also considered [redacted]

323 Peekya's consumer spend data are based on Google Play App-level data and include transactions through 2021, GOOG-PLAY-005535885 and GOOG-PLAY-010801689. See this report's production; https://support.google.com/googleplay/android-developer/answer/6112435 (describing a one-time $25 registration fee for developers on Google Play Store).

324 Peekya first published its app in December 2019. See this report's production.

higher for two-thirds to three-quarters of putative developer class members, depending on the year during the class period. Over the entire class period, about 67% of putative developer class members' fees would have been higher if Google charged [redacted] compared to the fees that they actually paid.[325] Contrary to Dr. Williams' assumption, those developers' costs would have been higher in the but-for world. Consequently, their profits would have been lower.

271. Similarly, if Google charged developers based on the number of downloads, as some Plaintiffs have suggested it should, some developers' service fees would have been higher, and their profits would be lower.[326] Generally, developers that have many downloads, but relatively low consumer spend would have higher service fees in a world where Google had charged for each download. For example, as described above, the developer [redacted] has an app with over [redacted] downloads and about [redacted] in consumer spend. If Google Play charged [redacted], [redacted] service fees would increase from about [redacted] (the actual service fees over the class period) to over [redacted].[327] Contrary to Dr. Williams' opinion, LowTech Studios' profits would not be higher in the but-for world.

272. Alternative Google Play monetization strategies would affect not only the service fees of developers, but also the kinds of services and tools available to developers. In the actual world, Google made investments in enabling subscription services, enabling different forms of payment, and others. In the but-for world, if Google was generating revenue in some other way, Google would likely refocus those investments towards services and products that are optimal for a but-for world monetization strategy. The proposed developer class members that rely on these tools and services may not get the same tools and services in the but-for world from Google. Whether they could have obtained the same tools at the same cost depends on the tools needed by the developer and their cost from other sources. For instance, developers that earn revenue from IAPs could have been worse off than developers that earn revenue from ads, for example, because Google could have invested more in tools for ads than tools for IAPs.[328]

VIII. CONSUMER PLAINTIFFS' EXPERT'S CLAIMS OF IMPACT ARE BASED ON FLAWED ECONOMIC REASONING AND DO NOT CONSIDER ACTUAL APP, SUBSCRIPTION, AND IAP PRICING

273. To show that any individual consumer was impacted by the alleged conduct, Dr. Singer, Consumer Plaintiffs' expert, must establish that (1) the developers from which the consumer purchased apps, subscriptions, or IAPs would have obtained lower service fee rates in the but-for world and (2) that as a result, the developers would have lowered their app,

---

325 See Exhibit 11.

326 See Developer Complaint ¶ 210; GOOG-PLAY-000336574 at 588.

327 The calculation assumes that the [redacted] See GOOG-PLAY-000336574 at 588; GOOG-PLAY-006990552 at 554, 565. See this report's production.

328 See Marchak Dep. Ex. 375 ("Because of a choice 12 years ago to charge 30% for premium downloads, we've invested in IAP, FOP coverage, DCB, Gift Cards, Growth Consulting, Subs, Points. Imagine if we decided to charge 30% of all ad sales instead?").

subscription and IAP prices. Unless both of those conditions occur, the consumer is not impacted. It follows that to show that all or nearly all consumers were impacted, Dr. Singer must establish that (1) all or nearly all developers would have obtained lower service fee rates in the but-for world and (2) those developers would have passed through those lower service fees in the form of lower retail prices to all or nearly all consumers.

274. Dr. Singer attempts to establish common impact by claiming that in the but-for world, competition from a single app store with a catalog of apps comparable to the catalog of Google Play would reduce Google Play's market share to 60% and that as a result, service fee rates for all paid apps would fall to [redacted] and service fee rates for all IAPs and subscriptions would fall to [redacted].[329] Alternatively, he claims that if paid apps and IAPs are considered together, all service fee rates, on all transactions, would be [redacted].[330] Dr. Singer reaches these results based on models that assume Google Play would set a certain service fee rate to maximize its profits on a portion of its sales. For example, Dr. Singer finds the but-for service fee rate of [redacted] based on a model that assumes Google maximizes profits on paid downloads. The model is constructed to return a single but-for service fee rate, as are his other models. All his models depend on average or aggregated inputs. Dr. Singer makes no attempt to study or test whether, in the but-for world, different developers would have been affected differently, and his models are not capable of explaining why some developers in the actual world get lower rates.

275. Dr. Singer further claims that nearly all developers would pass through nearly 100% of all service fee rate reductions to consumers. He finds that on average, 89.9% of the service fee reductions would have been passed through to consumers in the form of lower retail app and IAP prices.[331]

276. Dr. Singer's calculations of but-for service fee rates depend on his estimates of pass-through rates. If Dr. Singer's pass-through rates are wrong, then the but-for service fee rates that depend on the pass-through rates are also wrong. Of course, in addition, if the pass-through rates are wrong, even if there were service fee rate reductions, there would be no evidence that those reductions would have been passed through to consumers. Given the importance of Dr. Singer's pass-through rates, I begin with that topic in Section VIII.D. Discussion of the but-for service fee rate calculations are discussed in Section VIII.E.

D. Consumer Plaintiffs' Expert's Opinion Regarding Pass-Through Is Flawed and Inaccurate

277. Dr. Singer's conclusion of "widespread pass-through" is based on a formulation of pass-through rates that depend solely on developers' shares of unit sales within a "category."[332]

---

329 See Singer Report Tables 3, 5, ¶177.

330 Singer Report Appendix 4.

331 See Singer Report Table 8, ¶33. Dr. Singer also claims that in the but-for world Google could expand its Play Points program rather than reduce service fees. That opinion is discussed in Section IX below.

332 Singer Report at ¶239 ("The logit demand system yields a pass-through rate equal to [M-Q_j]/M, where M is the size of the market – inclusive of the outside good – and Q_j is the quantity sold of a given product.").

The method is overly simplistic, depends on unsupported assumptions, and includes no analysis of any service fee rate change or any cost change at all. Moreover, the method produces false results – that is, the method predicts pass-through rates of near 100% for many prices that did not change at all when the service fee rate changed.

278. Below, I first describe Dr. Singer's formula for finding pass-through rates and then show that the predicted pass-through rates based on the formula are contradicted by the empirical evidence in this case.

1. Dr. Singer's Pass-Through Methodology is Based on an Overly Simplistic Formula That Produces Inaccurate Results

279. Every pass-through rate calculated by Dr. Singer is calculated with a formula that depends solely on the share of a developer's sales within a "category." The "categories" used by Dr. Singer, which are integral to the results, are not based on any economic analysis or reasoning but are simply the categories used in Google Play. The method is unrealistic and produces pass-through rates that are demonstrably wrong.

280. An example demonstrates that Dr. Singer's method is based on overly simplistic and unrealistic assumptions. Dr. Singer finds that the pass-through rate for the developer [redacted] for subscription transactions in January 2019 is nearly 100%.[333] To find this pass-through rate, Dr. Singer calculated the ratio of [redacted] unit sales of its subscriptions transactions in that month – 2 transactions – and the total number of subscription transactions in the "Game" category in that month – 935,919 transactions. This ratio is [redacted] share of transactions (or unit sales) in that month in that category, which is 0.0002%.[334] According to Dr. Singer's method, the pass-through rate for [redacted] subscription transactions in that month is calculated as one minus the share, which is 99.9998%.[335] Dr. Singer performed similar calculations for other months and found [redacted] had a pass-through rate of 99.98% to 99.9999% throughout the class period.[336]

281. Dr. Singer's pass-through results are demonstrably wrong. Some developers, including [redacted], experienced service fee rate reductions in the real world, so data are available to test whether developers passed through service fee rate reductions consistent with Dr. Singer's predictions.

282. [redacted] service fee rate in Google Play fell from 30% to 15% in January 2018.[337] [redacted] service fee rates and its subscription prices are shown in Figure 10 below. The

---

333 [redacted] app is [redacted] identified as app package name [redacted] in the data.

334 That is, 2/935,919 = 0.0002%

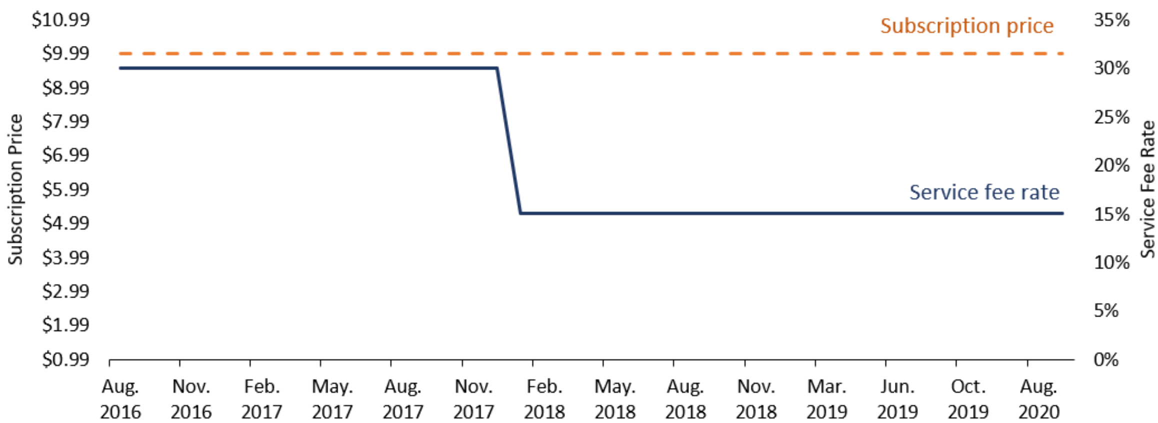
335 See Exhibit 44.

336 See Exhibit 45.

337 This rate reduction was pursuant to Google Play's policy change in 2018 for subscriptions for consumers that were more than one-year in length. When Google initiated the program and throughout the period [redacted] sold subscriptions in its [redacted] app, its consumer subscriptions apparently are annual renewals. Thus, its service fee rate dropped from 30% to 15% immediately after Google made the

graph shows that [redacted] did not change its retail price after its service fee rate reduction – its pass-through rate was zero. Dr. Singer finds the pass-through rate was nearly 100%. [redacted] did not pass on any of its service fee rate reduction, much less pass through nearly 100% of the reduction, as Dr. Singer's pass-through method predicts.[338]

Figure 10. [redacted] App Subscription Price and Service Fee Rate Contradicts Dr. Singer's Results of Nearly Complete Pass-Through



Source:

Exhibit 46.

283. Dr. Singer uses the same method to find pass-through rates for each developer in each app category and each month and for each of the three types of transactions – paid downloads, subscriptions, and IAPs. Each pass-through rate is calculated the same way – one minus the developer's share of unit sales in the category. The pass-through rates he reports for categories of apps[339] are weighted averages of these share-based pass-through rates.[340]

284. Across all developers, transaction types and months, 95% of pass-through rates calculated by Dr. Singer are over 99.5%.[341] Given that Dr. Singer's pass-through rates are

---

service fee rate reduction and continued to be 15% throughout the time it sold subscriptions. See this report's production.

338 Dr. Williams also found that [redacted] prices did not change after a service fee rate reduction. See Dr. Williams' backup for Williams Report Figure 3.

339 Singer Report Table 8.

340 The weights used in the category-weighted average are based on units sold. So, for example, for a category, there may be thousands of different pass-through rates calculated since Dr. Singer calculates a pass-through rate for three types of transactions, many months, and many developers. Each of those pass-through rates is weighted by its share of total units sold in the category, for each type of transaction in the month to find the category-weighted average pass-through rate.

341 See Exhibit 47.

determined solely with shares and that in his (overly broad) categories, there can be hundreds or even thousands of different apps and developers,[342] it is not surprising that many shares are very small and, consequently, when a pass-through rate is calculated as one minus the share, the resulting pass-through rates are very close to 100%.

285. The only way that Dr. Singer would conclude that some consumer was not impacted – because some amount of a service fee rate reduction was not passed through to the consumer – is if the consumer solely made purchases from developers that had 100% shares of their respective category sales.

2. Pricing Data Show that Dr. Singer's Predicted Pass-Through Rates Are Wrong

286. Dr. Singer's mistaken claim about [redacted] pass-through rate, described above, is not an isolated example. There are other examples of developers that Dr. Singer claims have pass-through rates of nearly 100% but in fact have had service fee rate reductions and have not passed through any part of the reduction in lower service fee rate reductions.[343]

287. Similar to [redacted], two other putative developer class members, [redacted] and [redacted], had service fee rate reductions from 30% to 15% in January 2018. As shown in Figures 11 and 12 below, these developers' retail prices remained constant after the service fee rate reduction, indicating a zero pass-through rate. Yet, Dr. Singer finds nearly 100% pass-through rates for both [redacted] and [redacted], for most months of the class period. Dr. Singer's pass-through rate formula produces inaccurate results for paid downloads and IAPs, as well.[344]

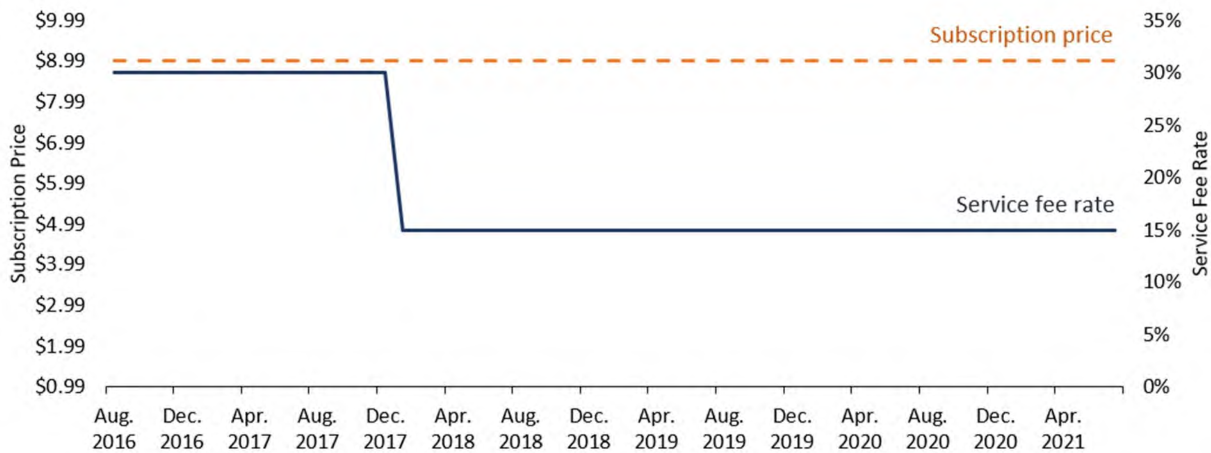
---

342 In the Games category used by Dr. Singer, there are 198,452 apps. See this report's production.

343 Dr. Singer's pass-through rates are also contradicted by testimony. He found a pass-through rate of nearly 100% for the developer class representative, Rescue Pets. But according to Mr. Scalise, of Rescue Pets, when it experienced a lower service fee rate, [redacted].

344 Most SKU prices did not change following a service fee rate reduction. See Figure 13 and Exhibit 52. These SKUs include not only subscriptions but also paid downloads and IAPs, such as the paid download app [redacted] (app package name [redacted]) and IAPs in [redacted] (app package name [redacted]). That is, [redacted] and [redacted] demonstrate zero pass-through even though Dr. Singer's method predicts a pass-through rate ranging from 99.98% to 99.996% for the former and 99.8% to 99.99% for the latter. See this report's production. Additional examples of prices that do not change after a service fee rate reduction are shown in Exhibits 63 and 64. Exhibit 63 shows the price and service fee rate for a subscription SKU, [redacted] The service fee rate for that SKU fell from 30% to 15% between July and September 2017, but the price remained constant from December 2016 through July 2021 – indicating no pass-through of the service fee rate reduction. Dr. Singer predicts pass-through rates applicable to this SKU that range from 81% to 100% (varying across months), based on his "one-minus-the-share" formula. Exhibit 64 shows the price and service fee rate for the [redacted] subscription SKU. The service fee for this SKU fell from 30% to 15% over the period December 2017 to December 2018; but the price of the SKU remained constant – indicating no pass-

Figure 11. [redacted] Monthly Subscription Price and Service Fee Rate Contradicts Dr. Singer's Results of Nearly Complete Pass-Through



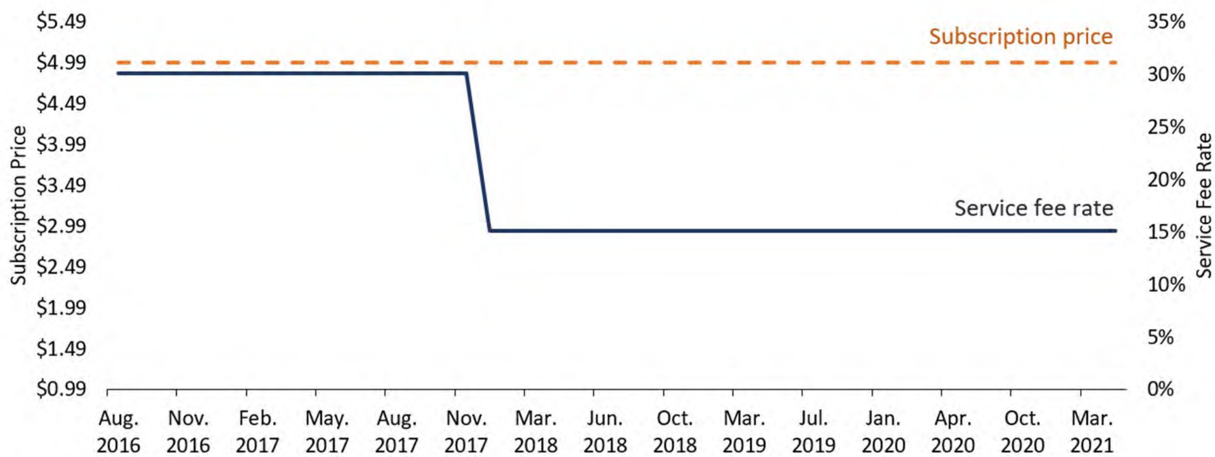
Source:

Exhibit 48

through of the service fee rate reduction. Dr. Singer predicts pass-through rates applicable to this SKU that range from 71% to 94% (varying across months), based on his "one-minus-the-share" formula.

HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY

Figure 12. [redacted] Monthly Subscription Price and Service Fee Rate Contradicts Dr. Singer's Results of Nearly Complete Pass-Through



Source:

Exhibit 49

288. Broadly considered comparisons of retail app, subscription, and IAP prices before and after service fee rate reductions indicate that only some rate reductions are passed through. These comparisons also show that Dr. Singer's method and pass-through results are not reliable and that his claim of "widespread" pass-through is not true.[345]

289. Figure 13 below summarizes comparisons of retail app and IAP prices before and after service fee rate changes and compares those results to Dr. Singer's pass-through rates based on his "one-minus-the-share" formula. The figure shows percentages of positive pass-through rates found by Dr. Singer, Dr. Williams, and me.

290. Dr. Singer predicts a pass-through rate for an individual developer-transaction type-app category-month based on his "one-minus-the-share" formula.

291. Dr. Williams compared prices of "SKUs" (e.g., prices of individual apps or IAPs)[346] that, based on Google's transaction data, had service fee rate changes from 30% to 15%. Dr. Williams compared the average price in all consecutive months in which the service fee rate was 30% and the average price in all consecutive months in which the service fee rate was 15%.[347] Dr. Williams made such comparisons for subscriptions, IAPs, and paid downloads.

345 Singer Report Section V.D.3.

346 SKU here refers to the "product ID" in Google's transaction data.

347 Williams fn. 119.

He found that for 23% of the paid downloads, 4% of the subscriptions, and 10% of the IAPs, prices fell after the service fee rate (that is, pass-through rates were positive).[348]

292. I compared prices of SKUs before and after the SKU had a service fee rate change. I analyzed SKU prices with the transactions data,[349] the app-level data, data that I collected once each month from Google Play between April 2021 and January 2022 ("scraped data"), and data requested by me from Google showing prices on June 21, 2021, October 16, 2021, and February 6, 2022 ("snapshot data"). I analyzed app prices, subscription prices, and IAP prices before and after those SKUs had a service fee rate change.

293. With Google's transaction data, I compared subscription prices one month before and one month after a service fee rate change as well as six months before and six months after the rate change.[350] I also separately examined subscription prices for developers in Google's LRAP, LRAP++, ADAP, and SwG programs (which I call "deal developers").[351]

- The app-level data extends through 2021 and therefore allowed me to consider price changes after, in July 2021, Google reduced the service fee rate for developers' first $1 million of consumer spend. With the app-level data, I compared prices of paid downloads one month before and one month after their service fee rate changed. I limit my analysis of the app-level data to paid downloads, because that data aggregates subscriptions and aggregates IAPs and thus could show changes in the average price due to changes in product mix. That problem does not exist for paid downloads. I analyzed paid app downloads that, according to the app-level data, experienced a service fee rate change in or after July 2021.
- The scraped data included prices of paid downloads only.[352] With the scraped data, I compared prices of paid downloads one month before and one month after their service fee rate changed. I identified paid downloads with service fee rate changes on or after July 2021 based on the app-level data.
- With the snapshot data, I compared prices of IAPs on June 21, 2021 (prior to Google Play's July 2021 rate change) to prices on October 16, 2021 and February 6, 2022 (after Google Play's July 2021 rate change). I included only IAPs that, according to the app-level data, experienced a service fee rate change on or after July 2021.

---

348 Overall, these percentages indicate about 8% of the SKUs' prices fell after the service fee rate was reduced. The percentage is consistent with Dr. Williams' Figure 3. Dr. Williams' analysis of paid download and non-subscription IAP SKUs relied on comparisons of prices before and after the July 2021 Google service fee rate change. Dr. Williams relied on the transactions data for these comparisons, which means he had only three days in July 2021 after the service fee rate change. App-level data was available for the full year of 2021 and could have been used for comparisons.

349 Dr. Williams used the same transactions data that I used.

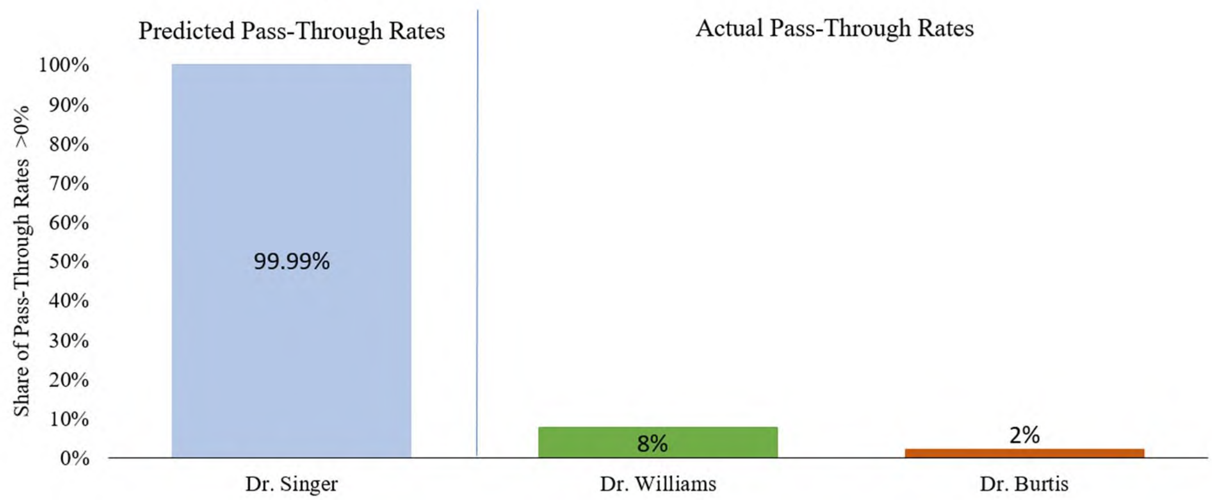
350 See Exhibit 50.

351 See Exhibit 8 for a list of the developers associated with these programs.

352 The paid download prices were the only prices that could be collected through the mechanical process that used to create the scraped dataset.

294. Figure 13 below summarizes the overall pass-through rate results from Dr. Singer, Dr. Williams, and from the analysis I conducted. The figure shows that while Dr. Singer's "one-minus-the-share" formula predicts that nearly all developers would pass through service fee rate changes, in fact, very few developers reduced prices when their service fee rates actually fell.[353]

Figure 13. Summary of Pass-Through Rates Found by Dr. Williams, Dr. Singer, and Dr. Burtis – Share of Positive Pass-Through Rates



Source:

Exhibit 51

295. The price comparisons before and after a service fee rate change summarized in Figure 13 – based on the comparisons I conducted and those provided in Dr. Williams' report – provide information about actual pricing decisions made by developers. Those results contradict Dr. Singer's formula-based conclusion of "widespread pass-through" and demonstrate that the method he used to calculate pass-through rates is not reliable. The results show that Dr. Singer's predicted pass-through rates nearly always show nearly complete pass-through, but when the pricing data are examined, those predicted pass-through rates are shown to be wrong and, in many cases, prices do not change when service fee rates change.

296. Moreover, Dr. Singer had data to test the reliability of his method and his results, but he failed to do so. He has no analysis in his report that attempts to discover how any developer (or set of developers) responded to any change in a service fee rate or to any change in any type of cost in the real world – even though there are examples of service fee rate reductions in the past that were available for him to analyze in the Google transactions data.[354] Instead,

353 Exhibit 52 shows these results in more detail.

354 As described below, Developer Plaintiffs' expert Dr. Williams analyzed certain service fee rate reductions. See Williams Report at ¶¶76-88.

he relied on a formula that produced results that conflict with the actual pricing behavior of developers.

297. Dr. Singer's results could also have been tested by comparing prices of apps, subscriptions, or IAPs that are available in Google Play (with a fee to Google Play) with prices on developers' websites (with no fee to Google Play). Dr. Singer claims that such comparisons yield "implied pass-through rates."[355] In some instances, prices on Google Play are the same as on developers' websites. According to Dr. Singer's implied pass-through rates, this would indicate zero pass-through. However, Dr. Singer's formula-based method predicts dramatically different pass-through rates. Table 8 provides examples of the implied pass-through rates, based on a comparison of the price in Google Play versus the price on the developer's website and Dr. Singer's predicted pass-through rates based on his "one-minus-the-share" formula. The table shows AllTrails, an outdoor app offering a Pro subscription plan at $29.99 per year both in Google Play and on its website – which means its implied pass-through rate is zero. However, Dr. Singer's predicted pass-through rate is 99%.

Table 8. Comparison of Dr. Singer's "Implied Pass-Through Rates" and His Predicted Pass-Through Rates

[redacted]

Source:

Exhibit 53.

3. Dr. Singer's Pass-Through Method and Formula Are Not Consistent with Economics

298. Dr. Singer's pass-through rate formula is not consistent with economics. According to Dr. Singer's formula, pass-through depends solely on a developer's "share," implying that the pass-through rate for two apps (or developers) are the same simply because they had the same share of their category's sales – regardless of any other differences between them. There are economic studies, both theoretical and empirical, that show pass-through depends on numerous factors, including supply conditions (e.g., cost), demand and the curvature of demand, and competitive conditions.[356] Dr. Singer's pass-through rates do not consider any of those factors.

355 Singer Report at ¶243, Table 9.

356 For a survey of the literature see "Cost pass-through: theory, measurement, and potential policy implications, A report prepared for the Office of Fair Trading," RBB Economics, February 2014.

299. Dr. Singer's "one-minus-the share" formula is not consistent with his own description of the economics of pass-through. According to Dr. Singer, a change in Google Play's service fee rate is "mathematically equivalent" to a developer's increase in marginal cost.[357] To support this opinion, Dr. Singer presents (another) formula that shows the relationship between a developer's marginal cost that includes the effect of a service fee rate ("C*"), the developers' "other" marginal costs ("C"), and the service fee rate ("t"). This relationship is derived by Dr. Singer from an assumption that a developer will set prices to maximize profits. He presents the formula as:[358]

$C^* = C/(1-t)$

300. According to the formula (and Dr. Singer), if the service fee rate is 30%, the developer's marginal costs, including the service fee will be equal to 1.43 times the developer's other marginal costs.[359] Similarly, if the service fee rate is 15%, the developer's marginal cost including service fees will be equal to 1.18 times the other marginal costs.

301. Dr. Singer concludes that if a developer sets prices to maximize profits (which implicates marginal cost in price-setting), and if the service fee rate in the but-for world is lower, prices in the but-for world will be lower. Dr. Singer's opinion is thus: (i) a service fee can be considered a marginal cost such that a lower service fee implies lower marginal cost, (ii) when marginal cost changes, price changes – which shows pass-through or a service fee rate change.[360]

302. However, Dr. Singer's opinion as described above contradicts his "one-minus-the-share" pass-through rate formula and his predicted pass-through rates.

303. First, Dr. Singer's opinion described above establishes that a pass-through rate for a service fee rate change depends on more than a developer's share. Dr. Singer's formulation indicates that the pass-through rate should consider the developer's marginal cost – and importantly, should consider the developer's marginal costs *besides the service fee*. To understand this, consider a developer that has an app with no marginal costs of distribution – that is, once the app is created and offered to consumers, the app can be downloaded and used by consumers without further cost. In Dr. Singer's formula described above, this would mean C=0. If the service fee rate is 30%, the developer's total marginal costs (according to Dr. Singer's formula) will be 1.43 times zero – or zero. If the service fee rate is 15% (again, using the same formula), total marginal costs including the service fee will be 1.18 times zero – or zero. In fact, if the developer's "other" marginal costs are zero, for any change in the service fee rate, total marginal costs including the service fee will be zero. Using Dr. Singer's logic, if the "other" marginal costs are zero, a change in the service fee rate will not

---

357 Singer Report at ¶225.

358 Singer Report at ¶225.

359 This is found by using 0.30 in the formula for "t," and 1.0 in the formula for C.

360 It should be noted that the entire chain of logic assumes that the developer sets prices to maximize profits and there is no "stickiness" in price changes, due to, for example developers setting prices to end in "99." As described above, these are assumptions that are not true for all developers and determining whether the assumptions are true would require investigation into the strategies of individual developers.

produce a marginal cost change, and therefore there is no reason to change price. Pass-through is zero. This result is consistent with my discussion above in Section VI.B that describes reasons we would expect no pass-through for apps that have zero marginal cost. These economic principles are reflected in economic literature recognizing that changing tax rates on profits do not affect the volume of output or prices. To the extent that Consumer Plaintiffs are suggesting that service fees calculated as a percentage of prices (an ad valorem fee) operate like taxes, if a developer has no marginal costs, reducing the service fee will function like a reduction in tax on profit, which the literature recognizes will not affect prices.[361]

304. Dr. Singer's claim about the "mathematical equivalence" of a service fee rate and marginal cost also establishes that his "one-minus-the share" formula is not accurate. The "one-minus-the share" formula does not consider marginal cost – it depends solely on a developer's share. Dr. Singer's "one-minus-the-share" formula is wrong and is inconsistent with Dr. Singer's opinions about marginal cost.[362] For a developer with a small share (like many, according to Dr. Singer's analysis) and a zero marginal cost, the "one-minus-the-share" formula returns a pass-through rate close to 100%. However, under Dr. Singer's formulation of the service fee rate as a marginal cost, this developer's pass-through rate would be 0%.

305. Dr. Singer's attempt to argue that a service fee rate change is "mathematically equivalent" to a change in marginal cost exposes the fact that pass-through depends on

---

[361] That is, when costs are zero, a tax applied as a percentage of profits is the same as the service fee rate applied to a developer's revenues (or price). In that case, changing the tax rate does not affect the price. See e.g., Goode, Richard, "The Corporate Income Tax and the Price Level," *American Economic Review*, Vol. 35, No. 1, 1945, pp. 40-58 at p. 43 ("From this it follows that, in so far as producers are guided by rational considerations, a tax on net profits, at any rate less than 100 per cent, will not directly affect the volume of output. A tax on profits is not itself a cost of production in any usual sense, nor does it directly influence costs. The tax will reduce the amount which a firm can retain out of the profits added by successive increments of output; nevertheless, any unit which adds to profits before taxes will also contribute something to profits after taxes. The last unit produced by each firm, its marginal unit, will add nothing to profits and nothing to taxes. Thus it will be advantageous for a firm to push production just as far with the tax as it would be if there were no tax. No firm will find the tax it pays a reason for changing its output. Hence the schedule showing the total supply which would be placed on the market by all firms in response to various prices will remain unchanged. So long as demand continues as before, market price will not be altered."). Moreover, Dr. Singer's claim that "Google's take rate is economically analogous to a tax on developers" and the materials he relies on to support that claim do not consider the case of marginal cost being equal to zero. See Singer at ¶244, fn. 531. See also Foros, Ø., Kind, H.J. and Wyndham, T., "Tax-Free Digital News?" *International Journal of Industrial Organization*, 66, 2019, pp.119-136, Kind, H.J. and Koethenbuerger, M., "Taxation in Digital Media Markets," *Journal of Public Economic Theory*, 2018, 20(1), pp.22-39, Sand-Zantman, W., "Taxation in the Digital Economy," 2018m Working Paper for the Institut d'Economie Industrielle.

[362] Had Dr. Singer considered that (as he did in ¶225 of his report), he would have found a different pass-through rate formula that depends on a developer's marginal cost. That is, to determine a pass-through rate, the developer's marginal cost (whatever its amount) should be considered. Dr. Singer's "one-minus-the share" formula does not. Dr. Singer's failure to consider marginal cost in determining pass-through rates is a fundamental problem with his method for all developers, not just those with zero marginal cost.

marginal cost. Of course, had Dr. Singer utilized a formula that includes a developer's marginal cost, determining a pass-through rate would depend on measuring the developer's marginal cost, which is a highly individualized analysis. Additionally, even that formula depends on certain assumptions that may or may not hold for all developers, including for example that they set prices to maximize profits and that their price-setting does not involve prices that end in "99."

4. The Basis For Dr. Singer's Pass-Through Formula is Fundamentally Flawed

306. Dr. Singer's basis for his "one-minus-the-share" formula is an overly simplified version of a highly restrictive demand system which, for the purposes here, produces unrealistic results. Dr. Singer's method uses "one minus the share" as the pass-through formula because, according to him, app demand can be described by a particular type of demand system – a simplified version of a "logit demand system" – and a pass-through rate in such a simplified logit demand system is equal to one minus the share.[363] The "logit" demand model, while frequently used in economics for its relative simplicity, is not appropriate for determining pass-through rates for apps because the model imposes restrictions that are not realistic for an app demand system.

307. There are thousands of apps and there can be thousands of apps even in a single category. Apps are highly differentiated from one another.[364] Because demand for any app depends on its characteristics, as well as the characteristics and prices of related apps, and characteristics of consumers, specifying a reliable demand model would require estimating many thousands of elasticities. Despite these complexities, Dr. Singer chose a relatively simple demand system – the logit system. The restrictions imposed by that relatively simple system when

---

363 Singer Report at ¶236. Dr. Singer uses regression analysis to estimate a logit demand system for apps. The regression is overly simplified in that there is no explicit consideration of app characteristics or consumers attributes or the interaction of the two, for example. Instead, Dr. Singer simply included "fixed effects unique to a given App and purchase type." Singer Report at ¶236. He claims that he has validated the use of his logit demand system, and by extension, the pass-through rate formula by observing for nearly all app categories, "a statistically significant relationship between demand and price" and so, concludes that the regression model is consistent with "economic expectations." (Singer Report at ¶237). He then uses the "one-minus-the-share" formula for pass-through derived from the logit model of demand for all pass-through rates. Singer Report at ¶239. To be clear, Dr. Singer's pass-through rates are not calculated with any of the regression "output." All pass-through rates depend only on the share of unit sales within a category. His formulas do not include any own-price elasticity of demand for any app or any cross-price elasticity of demand for any app generated from Dr. Singer's logit demand regressions. Moreover, Dr. Singer's observation that the price coefficients in the logit regressions are negative is not sufficient to infer that the results are "consistent with economic expectations" and that the formula for pass-through based on the logit model is a proper way to determine pass-through. For example, the logit model of demand used by Dr. Singer assumes that all products within a category are substitutes and that cross-price elasticities (e.g., the magnitude of substitution) depend on the products' shares, so that the best substitute for all products in a category is the product with the highest share. If those assumptions do not hold, then the logit model and the logit model's pass-through rate formula would not apply. See, for example, Train, Kenneth E., "Logit," in *Discrete Choice Methods with Simulation*, Cambridge University Press, 2009, pp. 34-75.

364 See Section IV.A .

applied to this case produce results that conflict with common sense (such that a pass-through rate can be determined solely with the share of the app's unit sales).[365]

308. One of the underlying assumptions of Dr. Singer's demand model and pass-through rates is that all products within a Google Play category are substitutes. There is no economic basis for this assumption and the assumption does not reflect realistic substitution patterns across apps. It is clearly not true for all apps within the categories used by Dr. Singer.[366] For example, his model requires that all apps within the category "Games" are substitutes. But the "Games" category combines games for toddlers such as "Thomas and Friends" (described as a "fun, safe & interactive game play for children aged 2-7"),[367] the action game "Doom" (described as "Violence, Blood and Gore" for ages 17+),[368] Zynga's card game "Poker – Texas Hold'em,"[369] and home design game "Redecor."[370] Dr. Singer's category of Games clearly does not satisfy the underlying assumption of the logit model.

309. The arbitrary nature of Dr. Singer's categories affects the pass-through rates because the pass-through rate formula (flawed as it is) depends on the definition of the categories. If the categories are wrong, the pass-through rates are wrong because the formula depends on the categories.[371]

---

365 See Berry, Steven T. and Phillip A. Haile, "Foundations of Demand Estimation," in *Handbook of Industrial Organization*, Vol. 4, No. 1, Elsevier, 2021, pp. 1-62 ("Although demand presents challenges that are absent in many empirical settings, all the "usual" challenges remain. One such challenge is finding empirical specifications that are both (a) sufficiently flexible to avoid strong a priori restrictions on the results and (b) sufficiently parsimonious to permit practical application. In some markets the number of closely related goods can be large; consider, for example, the set of all new automobile models, all computer models, all mutual funds, or all residential neighborhoods in a given city. Because the demand for a given good depends on the characteristics and prices of related goods, a demand system with J goods has J2 price elasticities at each point. In many contexts, this will rule out even a linear specification of the demand equation. Thus, even in cases where nonparametric estimation would be possible in principle, in practice it will often be necessary to impose restrictions in order to obtain an empirical model that is practical for the data available. Unsurprisingly, <u>one can go too far in the pursuit of parsimony. Some of the simplest demand specifications (e.g., the CES, multinomial logit, multinomial probit) impose strong a priori restrictions on demand elasticities, and, therefore, on markups, pass-through, and other key quantities of interest that are at odds with common sense and standard economic models.</u>" Emphasis added).

366 Singer Report Table 8. The only apparent basis for Dr. Singer's use of the categories is that they are used by Google "to track user purchase activity." Singer Report at ¶34.

367 https://play.google.com/store/apps/details?id=com.budgestudios.googleplay.ThomasAndFriendsMagicalTracks

368 https://play.google.com/store/apps/details?id=com.bethsoft.DOOM

369 https://play.google.com/store/apps/details?id=com.zynga.livepoker

370 https://play.google.com/store/apps/details?id=fi.reworks.redecor

371 App categories are chosen by developers. See e.g., https://www.storemaven.com/academy/how-to-choose-an-app-category/ ("The category you choose needs to be relevant to your app, of course. But it

310. This problem can be illustrated by showing that pass-through rates change when the category definitions change. In Google Play, there are 17 sub-categories within the Games category and assigning apps to different sub-categories would lead to different results in Dr. Singer's model.[372] For example, [redacted] belongs to the Adventure Game subcategory. Dr. Singer's formula finds an average pass-through rate of 98% for [redacted] (across all purchase types and months) because [redacted]'s share of the Game category is 2%. [redacted] share of the Adventure Game subcategory is 33% and so Dr. Singer's formula would find an average pass-through rate of 67%. Similarly, Dr. Singer finds that the pass-through rate for [redacted] is 57% because its share of all Games is 43%. [redacted] belongs to the Casual Game subcategory. Its share of Casual Games is 74% and so would have a 26% pass-through rate based on its share of its subcategory.[373] Dr. Singer's results are based on arbitrary category assignments, and his results change dramatically if different category assignments are used.

311. Even if Dr. Singer's "one-minus-the-share" formula could find reliable pass-through rates (which it cannot), the diversity among apps and complex substitution patterns means that identifying more appropriate categories would require analysis of many apps and would likely find more narrowly defined categories than those used by Dr. Singer. Such an analysis would be required to ensure that substitutes are properly grouped together.[374] As categories narrow, pass-through rates calculated based on Dr. Singer's method would decline. This follows from the formula itself. For example, say an app has a 1% share of the Games category. Dr. Singer's formula produces a pass-through rate of 99%. But if that app is relatively unique and has only a single other app that is a close substitute, and if each of those apps have a 50% share of its "category," the pass-through rate (based on Dr. Singer's formula) would decline to 50%. If the app was substantially more successful than its close

---

should also enable your app to rank well in category charts. Users often discover new apps to download by perusing the charts. This means that high-ranking apps generally receive a lot of traffic and are able to generate more organic downloads...The category you choose for your app can affect its discoverability and conversion rates in both the Apple App and Google Play stores. The trick to choosing the right category for your unique app is balancing relevance with your competition."); see also https://support.google.com/googleplay/android-developer/answer/9859673 ("You can choose a category and add tags to your app or game in Play Console. Categories and tags help users to search for and discover the most relevant apps in the Play Store.").

372 See Exhibit 3 for a list of the sub-categories within Games. Sub-categories are for example, Action Games, Card Games, and Sports Games, as well as others. Even within a sub-category, apps can be quite differentiated. For example, two Adventure Games are Stormfall: Saga of Survival, which describes itself as "survive, explore, and master crafting and sorcery in this free-to-play survival MMORPG set in the high-fantasy Stormfall world," and Sonic-Forces: Running Battle, which describes itself as "Sonic the Hedgehog is back and running in this fast and cool multiplayer racing & battle game." https://play.google.com/store/apps/details?id=com.pacific.wildlands; https://play.google.com/store/apps/details?id=com.sega.sprint.

373 See Exhibit 54.

374 Determining how one should allow for demand substitution to "outside products" (that is, in this context, outside of the category) is important, can affect results, and requires analysis. See e.g., Sheu, Gloria and Charles Taragin, "Calibrating the AIDS and Multinomial Logit Models with Observed Margins," U.S. Department of Justice, Economic Analysis Group Discussion Paper, 2012.

rival, and its share of the "category" grew to 99%, the pass-through rate (based on Dr. Singer's formula) would decline to 1%. As the example demonstrates, the formula depends critically on finding the "right categories," but the only way to find such categories would be to investigate each app and to determine its close substitutes. That involves a highly complicated and individualized analysis.[375] Dr. Singer has performed no such analysis – instead, even though the most critical part of the formula is the definition of the categories, he performed no analysis at all and simply used the categories in Google Play, even though those categories have little to do with the relevant substitution patterns across apps.

312. Dr. Singer's use of an average pass-through rate for each category, and his overall pass-through rate for all categories mask the variation in pass-through rates – even using Dr. Singer's arbitrary method. For example, he reports a pass-through rate for the category of "Games" is 92.3%.[376] But weighted average pass-through rates, like those reported by Dr. Singer, for sub-categories of Games show that pass-through rates vary across sub-categories. For example, the average pass-through rate for Casual Games would be 52% and for Trivia Games it would be 58%, both much lower than the 92.3% pass-through rate Dr. Singer calculates for "Games" as a single category.[377] Of course, these sub-categories similarly mask variation. Dr. Singer's overall pass-through rate, which he uses as an input into his but-for service fee models, is a weighted average of all pass-through rates, for all sub-categories and all categories.

5. Dr. Singer's Predicted Pass-Through Rates Are Based on Assumptions that Are Not Reliable

313. The pass-through rates found by Dr. Singer do not reflect the reality that many developers use a pricing strategy to set prices that end in "99." As described above, developers use such price points to influence consumers' perceptions of price and thus affect sales.[378] When supply or demand conditions change, developers may not respond to those changes because they do not want to change their price to the next price that ends in "99."

---

375 Dr. Singer's pass-through rates are calculated at the app developer GAIA ID, app category (pooling all games to one category), and purchase type level in each month, but the logic still applies at the app level. Such an analysis is further complicated by the fact that some of the apps at issue in this case have substitution relationships with free apps, which are pooled into the "outside products" category by Dr. Singer but ignored in his pass-through rate calculation. To determine "true" substitution relationships free apps must be considered. Moreover, any econometric model designed to identify substitution relationships (and thus identify reliable categories) must contend with the fact that developers frequently set prices that end in "99," and infrequently change prices. That feature of this marketplace will require more careful analysis for the estimation of demand relationships and determining any reliable categories.

376 Singer Report Table 8.

377 See Exhibit 55; Singer Report Table 8.

378 See, for example, Stiving, Mark and Russell S. Winer, "An Empirical Analysis of Price Endings with Scanner Data," *Journal of Consumer Research*, Vol. 24, No. 1, 1997, pp. 57-67 at 57 ("Managers apparently set prices in a manner consistent with the premise that the last digit of a price has a significant impact on sales. Several surveys on what price endings managers actually use have been conducted, and all of these surveys support the premise that firms set prices to appear that they are just below a round number.") See also Schindler, Robert M. and Patrick N. Kirby, "Patterns of Rightmost Digits Used in

314. Dr. Singer's calculations do not consider this marketplace reality and instead imply that developers would change prices – penny by penny – in response to service fee rate changes. In fact, Dr. Singer's pass-through rates imply that nearly all developers that use this strategy would abandon it in the but-for world. For example, the developer [redacted] subscription price in the actual world was $4.99 as of June 2017.[379] Dr. Singer's method predicts that if [redacted] service fee rate fell from 30% to [redacted] in June 2017 and had a pass-through rate of [redacted] (as he claims), [redacted] would change its price to [redacted] and change its pricing strategy.[380]

315. Table 9 below shows that Dr. Singer's calculations imply that among the over 80,000 developers that use focal point pricing, nearly all of them would abandon that strategy for at least one of their paid apps, subscriptions, or IAPs in the but-for world.[381]

Table 9. Share of Developers Abandoning the "99" Pricing Strategy in But-For World, August 2016 – December 2020

Developers	
Count of Developers Using ".99" Pricing in Actual World	80,103
Count of Developers Abandoning ".99" Pricing in But-For World	79,984
Share of Developers Abandoning ".99" Pricing in But-For World	99.9%

Source:

Exhibit 56.

316. Dr. Singer's pass-through rates are also not credible because, in some instances, the rates change month to month as shares change month to month. Figure 14 shows Dr. Singer's pass-through rates for three developers with dramatic changes in pass-through rates over time.[382] Figure 14 shows that Dr. Singer's pass-through rates for [redacted] a

Advertised Prices: Implications for Nine-Ending Effects," *Journal of Consumer Research,* Vol. 24, No. 2, 1997, pp.192-201 at 193-194; and Anderson, Eric and Duncan Simester, "The Role of Price Endings: Why Stores May Sell More at $49 than $44," May 2000, at http://ssrn.com/abstract=232542.

379 The developer only sold subscriptions through one app, com.ionicframework.oddhunter938355, that belongs to the Lifestyle category, in the class period.

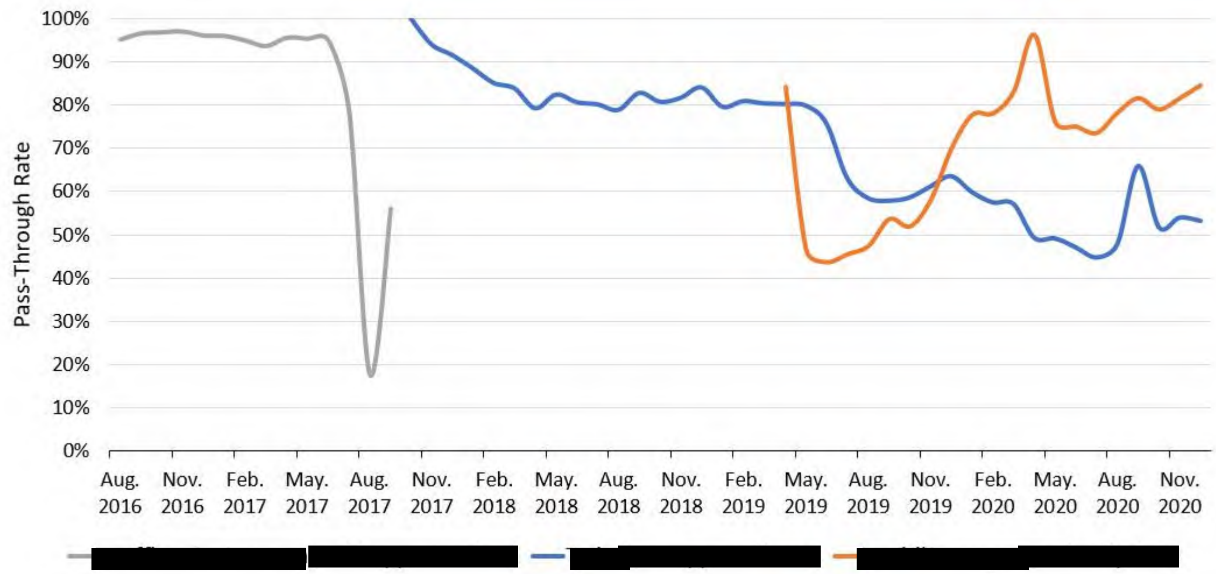
380 See Exhibit 57.

381 Table 9 includes developers for whom Dr. Singer calculated a pass-through rate that could be matched to the transactions data and who sold apps, subscriptions, or IAPs to US consumers only at list prices ending in "99."

382 Share changes of this magnitude are not common in Dr. Singer's calculations, although they do exist. The graph and discussion demonstrate the nonsensical nature of Dr. Singer's pass-through rate formula.

developer with IAP transactions apps in the Entertainment category, ranged from 44.8% to 99.98% over the period from October 2017 to December 2020. [redacted], another developer with subscriptions in the Travel & Local category, had pass-through rates that ranged between 43.8% and 96.1% over this period, and [redacted], an app with paid downloads in the Transportation category had pass-through rates of 17.8% to 96.9%, based on Dr. Singer's calculations.[383] The pass-through rates in Figure 14 fluctuate simply because the share of unit sales for these apps within their respective app category happen to change from month to month. If Dr. Singer's pass-through rates were accurate (which they are not), and if costs were constant over time, the prices of the apps, subscriptions, or IAPs would change dramatically month to month as the developer passed through different percentages of cost in its prices. In fact, however, there are no observed changes in prices in these apps over time.[384]

Figure 14. Pass-Through Rates for Three Developers, Based on Dr. Singer's Calculations, August 2016 – December 2020



Source:

Exhibit 58.

6. Given the Circumstances of this Case, Economics and Empirical Data Consistently Show that Determining Pass-Through Rates Requires an Individualized Analysis

---

383 See this report's production.

384 IAPs offered by [redacted]. have list prices ranging from $0.99 to $299.50 and have service fee rates ranging from [redacted] all remaining constant in the class period. The paid download of [redacted] has a list price of $1.51 and a service fee rate of [redacted], both remaining constant in the class period. Finally, the subscriptions offered by [redacted] have list prices ranging from $4.99 to $19.99 with no changes in the class period. As is the case for all subscriptions whose customers have been subscribed for at least one year, the service fee rate was reduced from 30% to 15% starting in 2018. See this report's production.

317. The economic factors that determine pass-through and the empirical data in this matter consistently show that determination of pass-through requires an individualized analysis.

318. Two of Plaintiffs' experts attempted to estimate "common" pass-through rates: Dr. Singer and Dr. Williams. They employed different approaches, but both attempted to show that the pass-through rate can be estimated using evidence common to all developers. However, they came to opposite conclusions with regard to the magnitude of a common pass-through rate.

319. According to Dr. Singer, 95% of pass-through rates are greater than 99.5%, indicating nearly complete pass-through. Dr. Singer finds that 99.99% of pass-through rates are positive, indicating that virtually every developer passed through at least some service fee rate increases to consumers, and therefore nearly all consumers were impacted.[385]

320. In contrast, according to Dr. Williams, 92% of the pass-through rates he considered are zero or negative, implying that 92% of developers did not pass through any of the price increases to consumers.[386]

321. Figure 15 below illustrates the two sets of results found by Consumer Plaintiffs' and Developer Plaintiffs' experts – which are nearly opposite of one another.

Figure 15. **Shares of Positive Pass-Through Rates Found by Dr. Singer and Dr. Williams Are Nearly Opposite of Each Other**

Source:

Exhibit 59.

322. Obviously, both experts' analyses cannot be true. Based on the economic principles and the empirical evidence in this case, neither Dr. Singer's nor Dr. Williams' analysis is accurate because determining pass-through rates for apps, subscriptions, and IAPs requires an individualized analysis into the relevant supply and demand conditions as well as into the pricing strategies used by developers.

323. As discussed above, a variety of economic factors determine how service fee changes (in particular the fact that the service fee is assessed as a percentage of price) will be passed

385 See Exhibit 47.

386 Williams Report at ¶80.

through in retail app, subscription, and IAP prices, and those characteristics vary across apps and developers.

324. For those developers that set prices to maximize profits, a service fee change (where the service fee is assessed as a percentage of price) will be more likely to affect price if the marginal cost of the app is high.[387] With marginal cost equal to zero, Dr. Singer's own model shows that a change in the service fee rate will have no impact on price.

325. Some apps have a positive marginal cost. For example, apps that pay royalties based on usage of the apps, such as some music streaming apps and other apps that play music within the app are likely to have positive marginal costs. Apps that provide ongoing customer support such as some multi-player games that incur processing costs for communications across users, are likely to have positive marginal costs.[388] Other apps are likely to have marginal cost near zero. This is a well-recognized feature of this industry as well as other industries where the product can be created once and sold multiple times. If developers of those apps obtained a lower service fee rate in the but-for world, and if they set prices to maximize profits, there is a high likelihood that consumers that purchase their apps and IAPs would not have paid lower prices.

326. Determining which apps are in which group requires individualized analysis of the cost conditions of those apps.

327. Furthermore, service fee rate changes are also less likely to be passed through in retail prices if developers set prices to end in "99." That pricing strategy is highly prevalent in this case; 97% of transaction prices of apps, subscriptions and IAPs end in "99."[389] The strategy leads to a "stickiness" in retail prices and therefore even if a developer obtained a lower service fee rate in the but-for world, set prices to maximize profits, and had positive marginal cost, it may not have changed prices when the service fee rate changed.

328. Whether app and IAP prices respond to service fee rate changes will also depend on the competitive conditions faced by a developer and whether a developer's rivals change their app and IAP prices. Apps are highly differentiated products, and any one app may have few or many rivals.[390] Determining the competitive conditions for any app requires individualized analysis.

329. Determining whether a developer would reduce prices in response to a service fee rate change requires some understanding of the developer's price-setting strategy. Not all

---

387 Singer Report at ¶225. In particular, the equations $(P-C^*)/P = 1/E\_D$ and $C^* = C/(1-t)$ indicate that the change in price P in response to the change in service fee rate, t, depends on the level of C that captures marginal costs other than the service fee. This also assumes that either developers do not set prices to end in "99," or that if they do, a price change to a price that ends in "99" satisfies the profit-maximizing conditions.

388 It is possible that marginal costs could be negative, in the sense that some apps generate revenue through additional users from, for example, advertising. In a model where a developer maximizes profits and marginal cost is negative, a reduced service fee rate would lead to higher app, subscription or IAP prices.

389 See Exhibit 32.

390 See Section IV.A.

developers set prices to maximize profits and therefore the economic models and results that depend on profit-maximization are not relevant to all developers. [redacted][392] Given that developers have different strategies, and do not all set prices to maximize profits, inquiry into the strategies and what those strategies imply for price reductions in response to service fee rate changes is necessary to understand pass-through. That is an individualized inquiry.

330. While the Plaintiffs' experts for the two proposed classes reach conclusions that either nearly all pass-through rates are positive, or nearly all pass-through rates are zero, economic models and evidence regarding the marketplace indicate that determining a pass-through rate requires individualized analysis – including analysis about the marginal cost of the app, the available and closeness of substitutes for the app, whether the app developer follows a focal point pricing strategy, and the overall strategy of price-setting used by the app developer. Plaintiffs' experts do not account for any of these factors. Moreover, determination of damages for the two proposed classes is further complicated by their sizes as well as the numerous factors that would affect damages and vary across proposed class members. Plaintiffs' experts have not even attempted to account for such factors.

E. Consumer Plaintiffs' Expert's Claim that All Service Fees Would Have Been Lower Only Assumes Classwide Impact

331. Dr. Singer has three different but-for service fee models.[393] He has one model for paid downloads – the Android App Distribution model,[394] another for subscriptions and IAPs –

391 [redacted]

392 [redacted]

393 Singer Report at ¶167.

394 Singer Report Table 3.

the IAP model,[395] and a third that combines paid downloads and IAPs – Android App Distribution/IAP model.[396]

332. Each of Dr. Singer's but-for service fee rates models assumes that Google sets a single service fee rate to maximize the profits it earns from Google Play on those types of transactions.[397] For example, Dr. Singer's Android App Distribution model assumes that Google sets a single service fee rate to maximize its profits on paid downloads and in the but-for world responds to competition by lowering the service fee rate to all developers with paid downloads. This is notwithstanding that in practice – and in the data Dr. Singer relied upon – not all developers have the same service fee rate.

333. Dr. Singer's profit maximization models that assume a single service fee rate in the actual world and return a single service fee rate in the but-for world also conveniently results in common impact. Indeed, in each of Dr. Singer's models, he does not allow for more than one service fee rate. Dr. Singer's models, therefore, assume the very thing – whether impact can be proven on a classwide basis – that he purports to test.

334. As described above, each of Dr. Singer's three models for but-for service fee rates requires a pass-through rate. In each of his three models, Dr. Singer uses the weighted average of the pass-through rates he obtained with the "one-minus-the share formula." For all the reasons described above, Dr. Singer's pass-through rates, both individually and as an average, are flawed. Therefore, Dr. Singer's but-for service fee rates, which depend on these pass-through rates, are flawed.

335. The pass-through rates in Dr. Singer's but-for service fee rate models matter to the results. For example, there is empirical evidence that some developers did not change their prices after a service fee rate reduction – that is, some pass-through rates are zero. If Dr. Singer had used a zero pass-through rate in Android App Distribution model, the but-for service fee rate for paid downloads would have been the same as the actual rate. That is, had Dr. Singer used a zero pass-through rate (for which there is empirical evidence for some proposed consumer class members), he would have found no impact to developers that

---

395 Singer Report Table 5.

396 Singer Report Appendix 4. Dr. Singer also has a model that purports to find but-for Play Point value ("But-For Buyer-Side Platform Competition Model"). See Singer Report at Table 10. The Android App Distribution Model (Table 3), the Combined Android App Distribution/In-App Aftermarket Model (Table A4 in Appendix 4), and the Buyer Side Platform Competition Model (Table 10) all purport to be models of a platform where buyers (consumers) and sellers (developers) interact. However, in two of the models – the Android App Distribution Model and the Combined Android App Distribution/In-App Aftermarket Model only a but-for service fee rate is found while "prices" to consumers (e.g., promotions) are held fixed. In the Buyer Side Platform Competition Model, a but-for "price" to consumers is found but the service fee rate is held fixed. So, Dr. Singer arbitrarily fixes one side of the platform and finds an equilibrium price of the other side, while the prices on both sides should be part of the same equilibrium decision. This is acknowledged by Dr. Singer himself in his description of the Rochet-Tirole model he claims to apply. Singer Report at ¶181.

397 Dr. Singer's assumption that rates are set to maximize Google Play's profits on subsets of transactions or even all paid transactions is overly simplistic and does not consider that if that was Google's strategy, it would likely find some other way to monetize Google Play so that it charged developers who offer free apps on Google Play – over 93% of developers. See Exhibit 7.

offered paid apps. The same result is true for Dr. Singer's Android App Distribution/IAP model that combines paid downloads, subscriptions, and IAPs. If Dr. Singer had used a zero pass-through rate, the but-for service fee rate would have been the same as the actual rate.[398]

336. More generally, in each of Dr. Singer's three models of but-for service fee rates, different pass-through rates lead to different but-for service fee rates. The problem is that Dr. Singer has no reliable method to determine pass-through rates (or an average pass-through rate) and because a pass-through rate is an important factor in all of his models of but-for service fee rates, those models, and the results produced based on the models, are also not reliable.

1. Dr. Singer's Calculated But-For Rates Assume Classwide Impact

337. Each of Dr. Singer's models treats all developers as if they are all the same. None of the models considers or attempts to explain why developers in the actual world obtain different service fee rates or different services and none of the models attempts to determine whether rates or services in the but-for world would vary across developers. Each of the models is premised on the assumption that all developers in the actual world are the same and all developers in the but-for world are the same. As described above, this assumption is not true in the actual world and there is no reason that it would be true in the but-for world. In the actual world, developers offer different apps, with different values to app stores, and they rely on different business strategies that make existing app stores or distribution options more or less attractive to them.[399] Those differences have enabled some developers to negotiate lower rates and more services. Those differences across apps and developers would continue to exist in the but-for world. But Dr. Singer's models do not address these differences, the effect of those differences in the actual world, or how such differences would affect rates and services in the but-for world.

338. Dr. Singer's assumption is also contradicted by the fact that not all developers may have the same competitive options in the but-for world. At least 22% of putative class developers

---

398 Singer Report Table 3 and Appendix 4. Dr. Singer's model of subscription and IAP transactions (Table 5) similarly depends on a pass-through rate and given that the pass-through rate method utilized by Dr. Singer is flawed, those but-for service fee rates are flawed. In that model, a lower pass-through rate will lower the but-for rate service fee rate – the opposite effect found in the other two models. If the pass-through rate is zero in Dr. Singer's Table 5 model, the but-for service fee rate is [redacted], rather than his calculated but-for rate of [redacted]. The underlying reason for this contradiction is that in his model of subscriptions and IAPs only (Table 5), Dr. Singer assumes that Google Play acts only as a payment processor and Google Play maximizes profits without considering consumers or the value to developers of "matching" consumers and developers. See Singer Report at ¶206. There is no basis (or evidence) that this assumption holds for all developers or apps with subscriptions and IAPs. Moreover, the assumption is inconsistent with Dr. Singer's opinions related to Play Points. That is, Play Points not only can be used for subscriptions and IAPs (as well as paid downloads) but [redacted]. See Lim Dep. Ex. PX 110 at -787 ("GPP continues to drive strong buyer churn reduction and spend lift").

399 Singer Report at ¶205 (Dr. Singer acknowledges that service fee rates pursuant to LRAP and similar programs were 15% or lower); ¶148 ("Google has cut in half the take rate for subscriptions, which are not limited to mobile devices; for subscriptions the platform is not as important, and they could command a lower rate."); ¶148 ("Project Hug resulted in effective take rate reductions for the largest developers, including in the In-App Aftermarket.")

(those that sell only at a $0.99 price) may not have the same options to independently contract with payment processors as others. Those developers offer apps, subscriptions, and IAPs at low price-points and because many payment processors include a per-transaction fee as well as a per-dollar fee to process transactions, the effective service fee rate from these payment processors for those developers is greater than 30%.[400] Dr. Singer explicitly disregards this fact. According to Dr. Singer, "Although many of the benchmark take rates in Table 6 entail a fee layered on top of a percentage of revenue, I do not impose such a fee in the but-for world modeled above."[401]

2. Dr. Singer's Calculations Rely on Inputs That Are Either Averages or Unsubstantiated Estimates Assumed to Apply to All Developers

339. In each of Dr. Singer's three models of but-for service fees, Dr. Singer takes certain inputs, such as an average app price, an average elasticity, and an average pass-through rate, among others, and then uses those inputs, along with certain assumptions about how Google sets its service fee rate to find a pass-through rate. In each model, Dr. Singer performs calculations twice: once for what he describes as the "actual world – monopoly" and then again for what he describes as the "but-for world – competitive." The principal difference between the actual and but-for calculations is that in the "but-for world-competitive" version, Google Play is assumed to have a 60% market share, instead of the market share Dr. Singer claims it has in the actual world.

340. The models' reliance on average inputs, as well as the models' finding of an average but-for service fee rate does not address the issue of common impact and masks the existence of developers that would not get a lower service fee rate in the but-for world. The models "hide" the fact that in the actual world some developers get a rate lower than the average rate and are incapable of determining whether a developer in the but-for world would get a lower rate than the average but-for rate.

341. Dr. Singer's but-for service fee rate models depend on numerous averaged inputs. For example, Dr. Singer uses an average app price of $3.99 when actual paid app prices vary from $0.01 to $400.00; and Dr. Singer uses an average subscription and IAP price of $8.99 when subscription and IAP prices vary from $0.01 to $1,341.90.[402] Dr. Singer's calculations assume that consumers obtain a "negative price" from Google Play Points. As described below, even though Google Play Points are available to all consumers, only [redacted] participate in the program at all and no more than [redacted] redeem points in the class period.[403] He uses an average pass-through rate of [redacted], when his own predicted rates vary and there is empirical evidence that many pass-through rates are zero. He also constructs average elasticities for both consumers and developers.[404] Other inputs to Dr. Singer's models are based on

---

400 See Exhibit 31.

401 Singer fn. 485.

402 See this report's production.

403 See Exhibit 60; Exhibit 61.

404 See Singer work product and Tables 3, 5, and Appendix 4. That is, Dr. Singer's models are not regression analyses that utilizes prices (or service fees) for individual developers and simply return an

assumptions that Dr. Singer claims apply to all developers. For example, Dr. Singer assumes competition for developers is the same for all developers in the actual world and that the effect of the purported increase in competition will be the same for all developers. Dr. Singer's method inherently eliminates variation among developers to artificially manufacture impact, where it may not exist so that the absence of impact cannot be detected.

342. The results of Dr. Singer's models change if his average inputs are replaced by inputs that describe individual or particular sets of developers. As described above, Dr. Singer uses a single pass-through rate of [redacted], which is the average of his calculated pass-through rates across all developers. While Dr. Singer's estimates of pass-through rates are inaccurate and do not reflect developers' pricing decisions, even those inaccurate estimates vary from 0% to 100%.[405] Moreover, there is empirical evidence that some pass-through rates are zero. Changing only the pass-through rate in Dr. Singer's models changes the results.[406]

343. Another important input to Dr. Singer's calculations is the increase in competition in the but-for world, which is characterized as an assumed reduction in Google Play's market share.[407] In the Android App Distribution model (Table 3) and in the combined Android App Distribution/IAP model (Appendix 4), Dr. Singer assumes Google Play's market share is 100% in the actual world and would be 60% in the but-for world. In the IAP model (Table 5), he assumes Google Play's share is 97% in the actual world and would be 60% in the but-for world.

344. In the actual world, Dr. Singer's assumptions about Google Play's market share imply that developers and consumers have no, or very limited alternatives to Google Play. This is not true. Dr. Singer recognizes that certain developers had alternatives to Google Play and were able to obtain service fee rates lower than 30% and he describes the service fee rates obtained by these developers as a "reasonable approximation of the but-for take rate."[408] In addition, Dr. Singer recognizes that many consumers have alternatives to Google Play. According to *his estimates*, over half of Android mobile devices have alternative app stores including the Samsung Galaxy Store which he finds to be available on approximately 52% to 57% of Android devices.[409]

---

average result. Dr. Singer's method utilizes averages of prices (and other inputs) and returns an average of those averaged inputs – it is an average of many averages, all of which are assumed to be true for all developers.

405 Dr. Singer finds 282 pass-through rates that are zero because in a particular month, that developer's share of its category and transaction-type transactions is 100%. See Exhibit 47.

406 As discussed in the prior section, a zero pass-through rate would find no service fee rate overcharge in Dr. Singer's model of paid downloads and his model of combined paid downloads, subscriptions, and IAPs.

407 According to Dr. Singer's calculation, if Google Play's market share is lower, consumers' elasticity of demand for apps will be higher (more elastic) and the buyer-side take-rate elasticity will be higher (more elastic), and as described above, the service fee rate will be lower. See Singer Report Table 3.

408 Singer Report at ¶¶158, 205.

409 Singer Report Table 1.

345. Dr. Singer's assumption that Google Play's market share in the but-for world is 60%, in all three of his models, is based on AT&T's long-distance market share in 1993.[410] Dr. Singer provides no evidence that AT&T is a reasonable benchmark for Google Play, that the competitive conditions in long-distance during regulation reflect Google Play in the actual world, or that competitive conditions in long-distance years after AT&T's divestiture reflect app distribution or IAPs in the but-for world.

346. But even if AT&T in 1993 could be considered an appropriate benchmark for certain aspects of competition in the but-for world, for purposes here, the question is whether it is appropriate to assume that an increase in competition would affect all developers in the same way, with the same but-for Google Play market share – as Dr. Singer does.[411] The answer is no. Given the diversity among developers described above, Consumer Plaintiffs' claimed increased competition in the but-for world would affect developers differently, just as competition in the actual world has resulted in different service fees and different services to developers.

3. Dr. Singer's Models Find Any Change in Competition Leads to Lower Rates for All Developers

347. Dr. Singer's models assume that for *any* change in "competition" (as measured by some reduction in Google Play's market share), no matter how small, there would have been some reduction in the service fee rate for *all developers*.

348. Dr. Singer's models of but-for service fee rates imply that if, absent the alleged conduct, Google Play's market share fell by a single percentage point, the rate for all developers' IAPs would fall [redacted] points (to [redacted])[412] and the rate for all developers' paid apps would fall [redacted] (to [redacted]).[413]

349. Dr. Singer's conclusion that any change in Google's market share would lead to lower service fee rates is contradicted by evidence from the actual world. In South Korea, the ONE Store, according to Dr. Singer, has obtained 14.9% of payment volume.[414] Dr. Singer's model would find that in Korea, Google Play's service fee rates to all developers would have fallen significantly, but in fact, that has not occurred. Instead, Google Play has reduced fees for particular developers through programs like Project Hug and provided incentives for

---

410 Singer Report at ¶216.

411 Indeed, competitive conditions in long-distance are not reasonably reflected in AT&T's overall market share. For example, certain long-distance competitors were not available in all states, so AT&T's market share likely varied across geography. See https://www.sprint.com/companyinfo/history/timeline_02.html (showing Sprint was available in 19 states in 1993).

412 Singer Report Table 3. This is calculated by changing the "Buyer-side Product Price Elasticity" to the "Monopoly Elasticity" divided by 99%.

413 Singer Report Table 5. This is calculated by changing row [12], "Google Market Share in the But-For World," to 96%.

414 Singer Report at ¶¶196-198.

high-value (high-spending) consumers through its Play Points program.[415] Dr. Singer's models assume that an increase in competition would have common impact, but the models fail to capture the way that competition works and that a purported increase in competition in the but-for world would not be common.

4. Dr. Singer's Profit-Maximization Models Do Not Consider That Google Can Utilize Alternative Monetization Strategies

350. Each of Dr. Singer's models assumes that Google sets service fees to maximize profits for either a subset of transactions or all paid transactions. This does not consider that Google has alternative ways to maximize profits and respond to competition, besides lowering service fee rates.

351. As described above, in Section VI.D, Google's alternative monetization strategies for Google Play would lead some developers to have higher costs in the but-for world and if those costs were passed through to consumers, as Dr. Singer claims, higher prices to consumers. Other developers could be better off with an alternative monetization strategy for Google Play. If those developers passed through benefits to consumers in lower prices, those consumers would have been better off in the but-for world. However, determining which consumers were worse off and which were better off depends on the alternative monetization strategy that would have been adopted by Google Play, which developers would have benefited, and which would have been made worse off from each strategy, as well as determining which consumers who purchased apps, subscriptions, or IAPs from those developers would have been better off and those that would have been worse off.

IX. PLAINTIFFS' EXPERTS' OPINIONS REGARDING GOOGLE PLAY POINTS DO NOT SHOW CLASSWIDE IMPACT

F. Google Play Points Program

352. Play Points is a consumer program that Google introduced in the U.S. in November 2019.[416] Before that, Google introduced the program in Japan and South Korea. Play Points allow consumers to earn "loyalty points" from purchases in Google Play that can then be used as Google Credits, to make purchases in Google Play, or redeemed for special IAPs.[417]

353. Google Play Points are available, without cost, to any consumer that registers to receive Play Points, but the program was intended to target and tends to benefit [redacted]

---

415 Dr. Sibley claims that Project Hug "inhibited expansion by existing stores and deterred entry by new stores," Sibley Report at ¶241. Dr. Singer claims that Project Hug "foreclose[ed] the ability of developers to enter into lucrative exclusive contracts with other app stores." See Singer Report at ¶110. Neither expert denies that Project Hug developers obtained favorable terms from Google Play. Their opinions imply that absent Project Hug, which they characterize as exclusionary, the Project Hug developers would not obtain those favorable terms. But whether these developers would be better off in the Plaintiffs' but-for world – without Project Hug – would require individualized proof of the deals that they would negotiate with other app stores and the terms of those deals compared to those they actually obtained in their individual negotiations with Google Play.

416 See Deposition of Tian Lim, November 28, 2021, ("Lim Dep"), Ex. 110 (GOOG-PLAY-002653782.R).

417 Lim Dep. at p.169.

[redacted] – those consumers that [redacted].[418] One feature of the program, for example, is that consumers who spend more in Google Play earn more Google Play Points. There are at least four user "tiers," with each level requiring a higher level of consumer spend and providing more points per dollar of spend.[419] For example, "Bronze" has no minimum amount of spend and generates 1 point per dollar spent; while "Platinum" has a $3,000 per year minimum consumer spend and generates 1.4 points per dollar spent.[420] Play Points was, in part, a competitive response by Google to [redacted][421]

354. Google's Play Points program not only targets particular consumers but is attractive only to certain consumers. Not all consumers register for and participate in Google Play Points. Among the roughly [redacted] U.S. consumers that transacted in Google Play during the period the Google Play Points program existed through 2021, only about [redacted] participated in Google Play Points.[422] That is, [redacted] of U.S. consumers did not participate at all in the Google Play Points program even though the program was available to them.

355. An even smaller percentage of consumers redeemed Play Points that were earned. From November 2019 through July 3, 2021, there were at most [redacted] U.S. consumers – about [redacted] of U.S. consumers in this period – that redeemed Play Points either by making transactions directly with the Play Points or by exchanging the Play Points for Google Credits and making purchases. In sum, only a small percentage of consumers enroll in the program and an even smaller percentage redeem Play Points.[423]

G. Plaintiffs' Experts' Claims Related to Play Points Do Not Show Classwide Impact

356. Consumer Plaintiffs' and Developer Plaintiffs' experts claim that in the but-for world, Google would expand its Play Points program, offering consumers enhanced or expanded

---

418 Lim Dep. Ex. PX 109 (GOOG-PLAY-000356806.R) at p. 22 [redacted]). See also Lim Dep. Ex. PX 110 (GOOG-PLAY-002653782.R) at p. 17 [redacted]); Lim Dep. Ex. PX 133 (GOOG-PLAY-000365029). [redacted]. See Feng Dep. Ex. PX 532 (GOOG-PLAY-001284083.R) at p. 33.

419 See Lim Dep. Ex. PX 110 (GOOG-PLAY-002653782.R) describing a "Diamond" level.

420 See Lim Dep. Ex. PX 110 (GOOG-PLAY-002653782.R) and https://gadgetstouse.com/blog/2020/10/22/what-are-google-play-points-how-to-use/ There are other benefits, that are similarly scaled by tier, including points on movies and books and weekly prizes.

421 See e.g., Feng Dep. Ex. PX 538 (GOOG-PLAY-009903056) at p. 10; Feng Dep. Ex. PX 531 (GOOG-PLAY-000292207.R) at p. 7; Feng Dep. Ex. PX 532 (GOOG-PLAY-001284083.R) at pp. 6, 54.

422 See Exhibit 60.

423 See Exhibit 61. The [redacted] Play Points redemption rate for U.S. consumers is an upper bound estimate that assumes that all "stored value" transactions were Play Points redemptions, as Google's transaction data does not distinguish between different types of stored value transactions. Only [redacted] of U.S. consumers redeemed Play Points by making transactions directly with Play Points.

benefits.[424] Neither Developer Plaintiffs' nor Consumer Plaintiffs' experts provide any reasoning or evidence that even if Google expanded the Play Points program in the but-for world, either of the two classes would have experienced classwide impact. For the reasons discussed below, the effect of an expanded Play Points program would not be common for either class.

357. Dr. Singer describes Play Points as a subsidy or "negative price" set by Google to consumers for "accessing" Google Play and considers that Google could lower the service fee rate or as an alternative, increase Play Points, effectively lowering the prices of apps and IAPs directly to consumers. Dr. Singer claims that in the but-for world, because Google Play's market share would fall to 60% (based on AT&T's market share in 1993), consumers' "platform price" would fall from a negative $0.2 per transaction to a negative $0.77 per transaction.[425] This reduction in the "platform price" is assumed to occur through the expansion of Play Points.

358. Dr. Singer fails to consider that Google's Play Points program provides benefits to a relatively small percentage of U.S. consumers. Over the time period when the program was available, less than [redacted] of U.S. consumers participated in the program and only up to [redacted] of U.S. consumers earned and redeemed Play Points.[426] Such low redemption rates are not unusual and are not the result of any alleged conduct.[427] Dr. Singer offers no analysis to show that all consumers would even participate in an expanded Google Play Points program in the but-for world, much less obtain benefits from an expanded program.

359. Dr. Singer claims that Google would expand Play Points program substantially more than Dr. Williams claims Google would. According to Dr. Williams, in the actual world, redeemed Play Points accounted for about [redacted] of consumer spend and in the but-for world, the "median percentage reward" would have been [redacted].[428] Dr. Singer finds that in the actual world, Play Points redeemed accounted for about [redacted] of consumer spend, but claims that in the but-for world, the expansion would have been nearly [redacted][429] – nearly [redacted] times what Dr. Williams claims.

---

424 Williams Report at ¶¶96-97, Singer Report at ¶33.

425 Singer Report at ¶¶179, 184, 245, Table 10.

426 See Exhibit 61.

427 For example, credit card programs offer extensive rewards and still many consumers do not participate. Bureau of Consumer Financial Protection, "The Consumer Credit Card Market," September 2021, Figure 1 shows that despite availability of reward credit cards, in each segment some transactions are on non-reward cards. Dr. Williams acknowledges that redemption rates are "generally low, consistent with Google's experience." That is, the low redemption rate cannot be construed to be implicated in the alleged conduct and low redemption rates would persist in the but-for world. Williams Report at ¶103, fn.135. See also https://www.prnewswire.com/news-releases/customers-sitting-on-100-billion-of-unredeemed-loyalty-points-623828294.html ("more than one-fifth of program members have never redeemed.").

428 Williams Report at ¶99.

429 Singer Report Table 10.

360. Neither Dr. Singer nor Dr. Williams considers that developers participate in Play Points by offering discounts on IAPs. Those developers contribute to the funding of the Play Points program but also receive benefits in increased revenues. Depending on how an expansion of Play Points would occur, some developers may be worse off by such an expansion. Predicting the effects of an expanded Play Points program on developers, without considering these issues, is highly speculative. Dr. Singer's model of Play Points does not recognize that developers participate in the program and does not consider how changes in the program could affect developers – including potentially the service fees paid by developers.

X. CONCLUSION

361. For the reasons discussed above, I find that one cannot establish antitrust impact to members of the proposed developer class using proof that is common to the class given the many individualized issues specific to developers.

362. I also find that one cannot establish antitrust impact to members of the proposed consumer class using proof that is common to the class given the many individualized issues specific to consumers,

363. Highly individualized analyses would be necessary to determine whether any given member of either proposed class would have been better off in the but-for world, and if so, what that member's individual damages would be.

364. Based on my analysis, there is a reasonable probability that among putative developer class members, at least 22% likely would not have lower service fees in the but-for world because the costs of separately contracting with payment processors may exceed Google Play service fees. Those putative developer class members price only at $0.99 and the per transactions costs charged by many payment processors would raise the total costs of payment processing beyond 30%. In addition, there are 36% of putative developer class members who set prices of $1.99 or lower that may not have been impacted, depending on the price "mix" of these developers and the services beyond payment processing that are valued by these developers. For all of the reasons described in this report, the percentage of developer class members who cannot show antitrust impact or injury is likely even higher, given the many individualized issues that affect whether any given developer would have been better off in Plaintiffs' but-for world. These includes developers' individual pricing strategies, the potential for higher costs to developers, plausible changes to Google Play's monetization strategy in the but-for world, among others. Indeed, for reasons such as these, the percentage of uninjured class members are likely higher.

365. Furthermore, based on my analysis, it is likely that a small percentage of proposed consumer class members were impacted. The available empirical evidence shows that among those developers that had a service fee rate reduction, a small percentage (e.g., between 2% and 8% depending on transactions analyzed by Dr. Williams and me) passed through service fee rate reductions in lower prices. In addition, 16% of proposed consumer class members spent $5 or less in Google Play during the entire class period. If, in the but-for world, those consumers had to pay more for free apps or to protect themselves from security issues, those consumers likely would have been worse off in the but-for world. Some 11% of the proposed consumer class used DCB. If in the but-for world service fee rates for DCB did not fall (because that form of payment is more costly) or if DCB was no

longer available, those consumers may not be impacted and could have been worse off in the but-for world. Finally, 71% of proposed consumer class members did not participate in Google Play Points, even though it was available to them. Therefore, even if in the but-for world that program was expanded, there is no credible evidence that such an expanded Play Points program would have benefited all proposed consumer class members.

366. Neither Consumer Plaintiffs' nor Developer Plaintiffs' experts provide a methodology, using common proof, that identifies which putative class members, or how many, suffered antitrust impact or injury because of Google's alleged conduct. Nor have they attempted to identify and exclude from the class those putative class members who may be worse off in a but-for world without Google's alleged conduct.

Executed on March 31, 2022 in Washington D.C.

____

Michelle M. Burtis, Ph.D.

APPENDIX A

Charles River Associates

Michelle M. Burtis
Senior Consultant

PhD, Economics
University of Texas
Austin

BA, Economics
University of Colorado
Boulder

Professional history

2020-present	*Senior Consultant*, Charles River Associates, Washington DC
2015–2020	*Vice President*, Charles River Associates, Washington, DC
2012–2014	*Senior Advisor,* Cornerstone Research, Inc., Washington, DC
2006–2011	*Vice President,* Cornerstone Research, Inc., Washington, DC
2004–2006	*Managing Director, Competition Policy,* LECG, Washington, DC
1998–2006	*Director,* LECG, Washington, DC
1988–1998	*Vice President,* Capital Economics, Washington, DC
1986–1988	*Associate,* ICF Incorporated, Washington, DC
1985–1986	*Lecturer,* University of Texas, Austin, TX

Publications

"Proving the Fix: Damages," with Keler Marku, GCR Guide to Private Litigation, 2019.

"Error Costs, Legal Standards of Proof and Statistical Significance," with Bruce Kobayashi and Jonah Gelback, *Supreme Court Economic Review, Volume 25, 2018.*

"Bringing Light to the DOJ"s Cartel Fine Methodology with the Auto Parts Plea Agreements," with Daniel K. Oakes and Mary Beth Savio, *Antitrust Law Bulletin*, 2018.

"The Effect of Optimal Penalties for Organizations Convicted of Price Fixing in the Presence of Criminal Sanctions for Individuals," with Bruce Kobayashi, Judge Douglas H. Ginsburg Liber Amicorum, 2018.

"Connecting the Dots: Tracing the DOJ's Inclusion of Indirect Commerce in Auto Parts Criminal Penalties," with Daniel K. Oakes and Mary Beth Savio, *Antitrust,* Summer 2018.

"Are Conspiracies to Manipulate Information Anticompetitive," with E. Buffmire, *Cartel and Criminal Conduct Newsletter*, ABA Section of Antitrust Law, Winter 2018.

"Regarding the Optimality of Cartel Fines," with Bruce Kobayashi, *Cartel and Criminal Conduct Newsletter*, ABA Section of Antitrust Law, 17:1- 22-30, Spring 2017.

"Economic Issues in Class Certification," in M Sanderson, S Moresi and D Sokol (eds.), *Antitrust Economics for Lawyers* (Matthew Bender & Co. 2017).

"Economic Analysis in Indirect Purchaser Class Actions," Who's Who Legal: Consulting Experts, 2016.

"Correlation and Regression Analysis in Antitrust Class Certification" with Darwin Neher. *Antitrust Law Journal,* 2011.

"Pass-Through, Common Impact, and Structural Modeling in Indirect Purchaser Class Certification" With Darwin Neher, Neill Norman, and Marshall Yan. *Economics Committee Newsletter, ABA Section of Antitrust Law,* 10:1, 3-8, Summer 2010.

"Petroleum Refining Industry Business Interruption Losses due to Hurricane Katrina." With Kivanc Kirgiz and David Lunin. *Journal of Business Valuation,* 2009.

"Economics of Antitrust: An Economic Analysis of Resale Price Maintenance." With D. Garrett and V. Howell. *The Antitrust Review of the Americas,* 2008.

"Modern Industrial Organization: A Comment." *George Mason Law Review,* Fall 2003.

"Vertical Integration and Agency Costs in the Gasoline Distribution Business." Working Paper presented at FTC Hearings on Gasoline Industry, June 2002.

"Why an Original Can Be Better than a Copy: Intellectual Property, the Antitrust Refusal to Deal and ISO Antitrust Litigation." With B. Kobayashi. *Supreme Court Economic Review,* 2001.

"Intellectual Property and Antitrust Limitations on Contract." With B. Kobayashi. *Competition in Dynamic Economics,* Cambridge University Press, J. Ellig, ed., 2011.

"Analysis of Alternative Methods of Forecasting Residential Housing Starts." Prepared for the US Department of Housing and Urban Development, 1988.

"Residential Energy Use in North Carolina." Prepared for the North Carolina Energy Conservation Group, 1987.

"Michigan Electricity Options Study, Socioeconomic Impact Analysis." Prepared for the Michigan Department of Energy, 1987.

"The Effects of Defense Spending on the Texas Economy: An Example of Concave Programming." With S. Thore and G. Kozmetsky. *Journal of Policy Modeling,* 1984.

Testimony, expert reports, and declarations

In Re Broiler Chicken Antitrust Litigation, Civil Action No. 1:16-cv-08637, 2022.

Reorganized FLI, Inc., v. The Williams Companies, Inc. and Dynegy Marketing & Trade, Case No. 2:05- cv- 02389- JAR – GEB, 2021.

In Re Peanut Farmers Antitrust Litigation, No.2:19-cv-00463, 2020.

Leaf Trading Cards, LLC v. The Upper Deck Company, No. 3:17-cv-3200- N. 2019.

First American Title Company, LLC et al. v. Title Data, Inc, et al. Cause No. 2016-69724; 2019.

Amphastar Pharmaceuticals, Inc. v. Momenta Pharmaceuticals, Inc., and Sandoz Inc. Civil Action No. 1:16-cv-10112-NMG; 2019.

Branta, LLC Harvest Holdings, Inc., v. Newfield Production Company, Civil Action No. 1:15- cv-00416- WYD-KLM; 2017.

In Re Western States Wholesale Natural Gas Antitrust Litigation, MDL Docket No. 1566; 2009, 2017.

In Re Cox Enterprises, Inc. Set-top Cable Television Box Antitrust Litigation, and *Richard Healy v. Cox Communications, Inc.*; 12-ML-02048-C, 2011, 2015.

Cap Carpet, Inc., v. Future Foam, Inc., et al., Case No. 13-CV-1140-JAR-KGG, 2015.

In Re Optical Disk Drive Products Antitrust Litigation, No 3:10-MD-2143 RS; 2013, 2015.

In Re Processed Egg Products Antitrust Litigation, MDL No. 2002; 2015.

Khurram Shah and Alpina Holdings Inc. v. LG Chem, Ltd., et al. in the Ontario Superior Court of Justice, CV-13-483540-00CP; 2014.

County of San Mateo v. CSL Limited, et al., 1:11-cv-01468; 2014.

In Re Polyurethane Foam Antitrust Litigation. Case No. 1:10 MD2196; 2013, 2014.

Karen McPeters et al. v. LexisNexis, Civil Action No. 4:11-CV-2056; 2014.

Presentations

"Disgorgement in the Public Interest." American Bar Association Antitrust Law Spring Meeting, Washington DC, 2013.

"Antitrust Economic Fundamentals." American Bar Association Antitrust Law Spring Meeting, Washington DC, 2008, 2009, 2010, 2011, 2012.

"Antitrust Law & Economics," Institute for Judges, Washington, DC, 2011.

"Market Definition and Market Power in Technology Markets." Antitrust Intellectual Property Conference, Berkeley, California, 2009.

"Expert Presentation for Mock Trial." American Bar Association Antitrust Litigation Course, Philadelphia, 2007 and Washington DC, 2009.

Other

ABA Long Range Planning Committee, 2018-2019.

ABA Cartel and Criminal Conduct Committee, Vice Chair and editor of newsletter, 2015-2018.

ABA Energy and Transportation Committee, Vice Chair and editor of newsletter, 2013-2015.

Editor, Market Definition, *Antitrust Law Developments*, 2003–2004.

Lecturer, Mathematical Economics, Graduate Department of Economics, George Mason University, 2003.

Appendix B

List of Materials Relied Upon

Academic Sources

Aguirregabiria, Victor, Allan Collard-Wexler, and Stephen P. Ryan, "Dynamic Games in Empirical Industrial Organization," in *Handbook of Industrial Organization*, Vol. 4, No. 1, 2021, pp. 225-343

Anderson, Eric and Duncan Simester, "The Role of Price Endings: Why Stores May Sell More at $49 than $44," May 2000, at http://ssrn.com/abstract=232542

Bar-Gill, Oren, "Algorithmic Price Discrimination When Demand Is a Function of Both Preferences and (Mis)perceptions," *University of Chicago Law Review*, Vol. 86, No. 2, 2019, pp. 217-254

Barron, John M. and Gerald J. Lynch, Gerald J., *Economics*, Third Edition

Berry, Steven T. and Philip A. Haile, "Foundations of Demand Estimation," in *Handbook of Industrial Organization,* Vol. 4, No. 1, Elsevier, 2021, pp. 1-62

Bizer, George Y. and Robert M. Schindler, "Direct Evidence of Ending-Digit Drop-Off in Price Information Processing," *Psychology & Marketing*, Vol. 22, No. 10, 2005, pp. 771-783

Chintagunta, Pradeep. "Investigating category pricing behavior at a retail chain." *Journal of Marketing Research*, Vol. 39, 2002, pp. 141-154

Evans, David S., "The Antitrust Economics of Multi-Sided Platform Markets," *Yale Journal of Regulation*, Vol. 20, 2003, pp. 325-381

Foros, Ø., Hans J. Kind, and Tim Wyndham, "Tax-Free Digital News?" *International Journal of Industrial Organization*, Vol. 66, 2019, pp.119-136

Ghose, Anindya, and Sang Pil Han, "Estimating Demand for Mobile Applications in the New Economy," *Management Science*, Vol. 60, No. 6, 2014, pp. 1470-1488

Goode, Richard, "The Corporate Income Tax and the Price Level," *American Economic Review*, Vol. 35, No. 1, 1945, pp. 40-58

Johnson, John and Greg Leonard, "Rigorous Analysis of Class Certification Comes of Age," *Antitrust Law Journal*, Vol. 77, 2011, pp. 569-586

Reference Manual on Scientific Evidence, National Academies Press, Third Edition, 2011, pp. 429-430

Kind, H.J. and Marko Koethenbuerger, "Taxation in Digital Media Markets," *Journal of Public Economic Theory*, Vol. 20, No. 1, 2018, pp.22-39

Kobayashi, Bruce H. and Joshua D. Wright, "What's Next in Apple Inc. v. Pepper? The Indirect-Purchaser Rule and the Economics of Pass-Through," *Cato Supreme Court Review*, 2018, pp. 249-269

Lambrecht, Anja, et al., "How do firms make money selling digital goods online?" *Marketing Letters*, Vol. 25, No. 3, 2014, pp. 331-341

Liobet, Gerard and Jorge Padilla, "The Optimal Scope of the Royalty Base in Patent Licensing," *Journal of Law & Economics*, Vol. 59, No. 1, 2016, pp. 45-73

McAfee, Preston R. and Vera te Velde, "Dynamic Price Discrimination in the Airline Industry," in T.J. Hendershott, *Handbook on Economics and Information Systems*, Vol. 1, Elsevier, 2006

Samuelson, Paul A. "Consumption Theory in Terms of Revealed Preference." *Economica*, Vol. 15, No. 60, 1948, pp. 243-253.

Sand-Zantman, W., "Taxation in the Digital Economy," Working Paper for the Institut d'Economie Industrielle, 2018

Schindler, Robert M. and Patrick N. Kirby, "Patterns of Rightmost Digits Used in Advertised Prices: Implications for Nine-Ending Effects," *Journal of Consumer Research, Inc.*, Vol. 24, No. 2, 1997, pp.192-201

Sheu, Gloria and Charles Taragin, "Calibrating the AIDS and Multinomial Logit Models with Observed Margins," U.S. Department of Justice, Economic Analysis Group Discussion Paper

Shiller, Benjamin Reed, "First-Degree Price Discrimination Using Big Data," Working Papers 58, Brandeis University, Department of Economics and International Business School, 2014

Stavins, Joanna, "Price discrimination in the airline market: The effect of market concentration," *The Review of Economics and Statistics*, Vol. 83, No. 1, 2001, pp. 200-202

Stiving, Mark and Russell S. Winer, "An Empirical Analysis of Price Endings with Scanner Data," *Journal of Consumer Research, Inc*., Vol. 24, No. 1, 1997, pp. 57-67

Train, Kenneth E., "Logit," *Discrete Choice Methods with Simulation*. Cambridge University Press, 2009, pp. 34-75

Weber, Thomas A., "Delayed Multi-Attribute Product Differentiation," *Decision Support Systems*, Vol. 44, 2008, pp. 447-468

Bates-Numbered Documents

EPIC_GOOGLE_00089350
EPIC_GOOGLE_01413875
EPIC_GOOGLE_01581798
EPIC_GOOGLE_03978760
EPIC_GOOGLE_03979041
EPIC_GOOGLE_03983189
EPIC_GOOGLE_03986544
EPIC_GOOGLE_04290360
EPIC_GOOGLE_04415004
GOOG-PLAY-000097630
GOOG-PLAY-000232540
GOOG-PLAY-000236162
GOOG-PLAY-000237766
GOOG-PLAY-000271389
GOOG-PLAY-000286913
GOOG-PLAY-000292207.R
GOOG-PLAY-000310705
GOOG-PLAY-000337564
GOOG-PLAY-000355655
GOOG-PLAY-000356806.R
GOOG-PLAY-000437819
GOOG-PLAY-000444214.R
GOOG-PLAY-000521845
GOOG-PLAY-000541836
GOOG-PLAY-000542244
GOOG-PLAY-000560397
GOOG-PLAY-000560564
GOOG-PLAY-000561051.R
GOOG-PLAY-000565001.R
GOOG-PLAY-000565172.R
GOOG-PLAY-000565541
GOOG-PLAY-000568027
GOOG-PLAY-000604733
GOOG-PLAY-001284083.R
GOOG-PLAY-001291192
GOOG-PLAY-001507601
GOOG-PLAY-001507602
GOOG-PLAY-002653782.R
GOOG-PLAY-003331592.R
GOOG-PLAY-003331764
GOOG-PLAY-003333689
GOOG-PLAY-003335685
GOOG-PLAY-003335786.R
GOOG-PLAY-005535885
GOOG-PLAY-005535886
GOOG-PLAY-006817773.R
GOOG-PLAY-006990552
GOOG-PLAY-006990552
GOOG-PLAY-006998204R
GOOG-PLAY-007203251
GOOG-PLAY-007335783
GOOG-PLAY-007335784
GOOG-PLAY-007335785
GOOG-PLAY-007335786
GOOG-PLAY-007335787
GOOG-PLAY-007335788
GOOG-PLAY-007335789
GOOG-PLAY-007335790
GOOG-PLAY-007335791
GOOG-PLAY-007335792
GOOG-PLAY-007335793
GOOG-PLAY-007335794
GOOG-PLAY-007628059
GOOG-PLAY-00737956
GOOG-PLAY-00737958
GOOG-PLAY-00737959
GOOG-PLAY-00737961
GOOG-PLAY-009903056
GOOG-PLAY-009908838
GOOG-PLAY-010569450
GOOG-PLAY-010569458
GOOG-PLAY-010801688

GOOG-PLAY-010801689
GOOG-PLAY-0110236932
GOOG-PLAY2-000483364
PSB-GOOGLE-0002255
SEA_GOOGLE_00002109
SEA_GOOGLE_00002117
SEA_GOOGLE_00002124

Court and Legal Documents

Cartwright Act (Cal. Bus. & Prof. Code ¶¶ 16700 et seq.)

Complaint, *State of Utah et al. v. Google LLC*, filed July 7, 2021, Case No. 3:21-cv-05227

Consolidated Second Amended Class Action Complaint, *In re Google Play Consumer Antitrust Litigation,* filed December 20, 2021

Epic v. Apple Transcript of Bench Trial Proceedings Day 2, May 4, 2021

Interrogatory No. 6, Peekya App Services, Inc.'s Responses and Objections to Defendants' First Set of Interrogatories to App Developer Class Plaintiffs, *In Re Google Play Store Antitrust Litigation*, November 11, 2021, Case No. 3:21-md-02981-JD and Case No. 3:20-cv-05792-JD

Response to Interrogatory No. 3, Pure Sweat Basketball, Inc.'s Responses and Objections to Defendants' First Set of Interrogatories, *In re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD, November 19, 2021

Second Amended Consolidated Class Action Complaint *In re Google Play Developer Antitrust Litigation*, filed January 21, 2022.

Testimony of Horacio Gutierrez Head of Global Affairs & Chief Legal Officer, Spotify U.S. Senate Judiciary Committee Subcommittee on Competition Policy, Antitrust, and Consumer Rights April 21, 2021

Unfair Competition Law (Cal. Bus. & Prof. Code ¶¶17200 et seq.)

Depositions (with Exhibits)

Deposition of Adam Sussman, January 7, 2022

Deposition of Alec Shobin, January 26, 2022

Deposition of Alex Iwamoto, February 18, 2022

Deposition of Andrew Grant, December 7, 2021

Deposition of Christian Cramer, January 13 and 14, 2022

Deposition of Christopher Babcock, February 17, 2022

Deposition of Daniel Egerter, January 20, 2022

Deposition of Daniel Scalise, March 11, 2022

Deposition of David Kleidermacher, February 3 and 4, 2022

Deposition of Ed Zobrist, December 16, 2021

Deposition of Eric Chu, December 20, 2021

Deposition of Hans Stolfus, February 11, 2022

Deposition of Haseeb Malik, March 4, 2022

Deposition of Jamie Rosenberg, February 10, 2022

Deposition of Jim Kolotouros, February 2 and 3, 2022

Deposition of Kevin Wang, December 15, 2021

Deposition of Kobi Glick, December 15 and 16, 2021

HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY

Deposition of Lacey Ellis, March 22, 2022

Deposition of Lawrence Koh, December 9, 2021

Deposition of Matthew Atkinson, March 18, 2022

Deposition of Mary Carr, December 21, 2021

Deposition of Mike Marchak, January 12 and 13, 2022

Deposition of Paul Feng, January 14 and 18, 2022

Deposition of Paul Gennai, January 18, 2022

Deposition of Richard Czeslawski, March 21, 2022

Deposition of Sameer Samat, February 2 and 3, 2022

Deposition of Serena Moglia, February 10, 2022

Deposition of Steven Allison, January 26, 2022

Deposition of Tian Lim, December 2, 2021

Deposition of Zachary Palmer, January 21, 2022

Expert Reports

Expert Report of Dr. David. S. Sibley, February 28, 2022, *In Re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD

Expert Report of Dr. Hal J. Singer, February 28, 2022, *In Re Google Play Consumer Antitrust Litigation*, Case No. 3:20-cv-05761-JD and Case No. 3:21-md-02981-JD

Expert Report of Dr. Michael A. Williams, February 28, 2022, *In Re Google Play Developer Antitrust Litigation*, Case No. 3:20-cv-05792-JD

Other Sources

All materials referenced in this report, its exhibits, and its production

"A Beginner's Guide to Mobile Apps," Lifewire, https://www.lifewire.com/what-are-apps-1616114

Amazon Developer Services Agreement, last updated September 25, 2018, https://web.archive.org/web/20181026122837/https://developer.amazon.com/support/legal/da

"Apps for Everyone," Windows, https://www.microsoft.com/en-ca/windows/windows-10-apps

Bank of America, https://promotions.bankofamerica.com/digitalbanking/mobilebanking

"Big Data and Personalized Pricing," 2015, Council of Economic Advisors Report, https://obamawhitehouse.archives.gov/sites/default/files/whitehouse_files/docs/Big_Data_Report_Nonembargo_v2.pdf

"Boosting developer success on Google Play," Android Developers Blog, March 16, 2021, https://android-developers.googleblog.com/2021/03/boosting-dev-success.html

Booty, Matt, "Continuing Our PC Gaming Journey 2021 and Beyond," Xbox, April 29, 2021, https://news.xbox.com/en-us/2021/04/29/continuing-our-pc-gaming-journey-in-2021-and-beyond/

"Century Games," LinkedIn, https://www.linkedin.com/company/century-game

"Cost pass-through: theory, measurement, and potential policy implications, A report prepared for the Office of Fair Trading," RBB Economics, February 2014

County of San Diego apps, https://www.sandiegocounty.gov/content/sdc/dmpr/gfx/appcenter/index.html

"Developer Economics," Roblox, https://developer.roblox.com/en-us/articles/developer-economics

Electronic Arts Inc. Form 10-K, 2020

European Commission Press Release, Antitrust: Commission sends Statement of Objections to Apple on App Store rules for music streaming providers, Brussels, 30 April 2021

"Evolving our business model to address developer needs," Android Developers Blog, October 21, 2021, https://android-developers.googleblog.com/2021/10/evolving-business-model.html

Fingas, Roger, "Apple announces it will offer App Store subscriptions to all apps, take smaller 15% cut," Apple Insider, June 8, 2016, https://appleinsider.com/articles/16/06/08/apple-announces-it-will-offer-app-store-subscriptions-take-smaller-15-cut

"Five Common Operating Systems," Small Business Chron, https://smallbusiness.chron.com/five-common-operating-systems-28217.html

"Google Play is lowering its developer fees for app subscriptions," XDA Developers, October 21, 2021, https://www.xda-developers.com/google-play-is-lowering-its-developer-fees-for-app-subscriptions/, accessed March 25, 2022

"Google Play Lowers App Subscription Fee to 15 Percent, Matches Apple's Offering," Gadgets 360, October 20, 2017, https://gadgets.ndtv.com/apps/news/google-play-app-subscription-fee-30-percent-to-15-1764923, accessed November 8, 2021

Kamen, Matt et. al, "Best Switch exclusives to make sure you have in your library," Gamesradar, accessed January 20, 2022, https://www.gamesradar.com/best-switch-exclusives/

"Listening to Developer Feedback to Improve Google Play," Android Developers Blog, September 28, 2020, https://android-developers.googleblog.com/2020/09/listening-to-developer-feedback-to.html

ONE store Introduction, ONE store Corp., January 2020, https://dev.onestore.co.kr/devpoc/static-res/files/ONEstoreIntro_dev_en.pdf

"Playtime 2017: Find success on Google Play and grow your business with new Play Console features," Android Developers Blog, October 19, 2017, https://android-developers.googleblog.com/2017/10/playtime-2017-find-success-on-google.html

Pure Sweat Basketball, https://web.archive.org/web/20170214231549/https://puresweatbasketball.com/training-app/

Roblox Corp. Form 10-Q, September 30, 2021, at pp. 10, 12-13, https://d18rn0p25nwr6d.cloudfront.net/CIK-0001315098/ad69fbb0-a7b7-465b-942f-53206ff42303.pdf

Samsung Galaxy terms and conditions 2014, https://seller.samsungapps.com/notice/getNoticeDetail.as?csNoticeID=0000002677

San Francisco taxi mobile apps, https://www.sfmta.com/taxi-mobile-apps

Sensor Tower, https://app.sensortower.com/android/rankings/top/mobile/us/overall

"Shop," LittleHoots, https://www.littlehoots.com/shop/p/ftxisgkwjypb7d6v7zgsxrae8saj8l, accessed January 26, 2022

Steamworks Development, Steam Community, November 30, 2018, https://steamcommunity.com/groups/steamworks/announcements/detail/1697191267930157838, accessed November 8, 2021

"Top Charts: US – Overall," SensorTower, https://app.sensortower.com/android/rankings/top/mobile/us/overall

"[Update: Now in effect] Google raises subscription revenue for providers from 70% to 85%, but only for users retained after 12 months," Android Police, January 2, 2018, https://www.androidpolice.com/2018/01/02/google-raises-subscription-revenue-providers-70-85-users-retained-12-months/, accessed November 10, 2021

"What Is an MMO?" Lifewire, https://www.lifewire.com/what-is-an-mmo-4687003, accessed February 24, 2022

"What Is Internet Security?" McAfee, https://www.mcafee.com/enterprise/en-us/security-awareness/cybersecurity/what-is-internet-security.html, accessed November 18, 2021

"What is Uptodown?" Uptodown, https://www.uptodown.io/what-s-uptodown, accessed November 17, 2021

"WSJ mobile and tablet apps." https://store.wsj.com/shop/us/us/wsjuelnsb20/, accessed February 11, 2021

U.S. federal government apps, https://www.usa.gov/mobile-apps

University of Connecticut, https://mobile.uconn.edu/applications/

Zynga Inc. Form 10-K, 2020

https://about.crazygames.com/

https://activision.helpshift.com/a/cod-mobile/?p=all&s=cod-points-credits-and-battle-pass&f=what-are-cod-points&l=en

https://activision.helpshift.com/a/cod-mobile/?p=all&s=cod-points-credits-and-battle-pass&f=what-is-battle-pass&l=en

https://bethesda.net/en/store/home

https://bookedin.com/appointment-booking-payment-system-plans-and-pricing/

https://developer.accuweather.com/packages

https://developer.amazon.com/docs/app-submission/migrate-existing-app.html

https://developer.amazon.com/docs/in-app-purchasing/iap-overview.html#what-is-in-app-purchasing-iap

https://developer.android.com/distribute/best-practices/launch/pre-registration

https://developer.android.com/distribute/google-play

https://developer.android.com/distribute/play-billing

https://developer.android.com/studio/publish

https://developer.android.com/topic/google-play-instant/overview

https://developer.apple.com/support/compare-memberships

https://developer.apple.com/programs/video-partner/

https://developer.apple.com/support/enrollment/

https://developers.google.com/android/play-protect/phacategories

https://devices.netflix.com/

https://docs.aws.amazon.com/lumberyard/latest/userguide/lumberyard-intro.html

https://en.aptoide.com/company/about-us

https://evernote.com/compare-plans

https://ezi.am/

https://gadgetstouse.com/blog/2020/10/22/what-are-google-play-points-how-to-use/

https://help.netflix.com/en/node/101653

https://help.pandora.com/s/article/Royalties-and-Spins?language=en_US

https://obamawhitehouse.archives.gov/blog/2015/02/06/economics-big-data-and-differential-pricing

https://play.google.com/about/developer-distribution-agreement.html

https://play.google.com/about/howplayworks/

https://play.google.com/console/about/acquisitionreporting/

https://play.google.com/console/about/app-bundle-explorer/
https://play.google.com/console/about/closed-testing/
https://play.google.com/console/about/devicecatalog/
https://play.google.com/console/about/internalappsharing/
https://play.google.com/console/about/internal-testing/
https://play.google.com/console/about/keymanagement/
https://play.google.com/console/about/monetize/
https://play.google.com/console/about/opentesting/
https://play.google.com/console/about/partnerprogram/
https://play.google.com/console/about/playgamesservices/
https://play.google.com/console/about/pre-registration/
https://play.google.com/console/about/stats/
https://developers.google.com/android/play-protect
https://play.google.com/console/about/store-listing-experiments/
https://play.google.com/console/about/storelistings/
https://play.google.com/console/about/translationservices/
https://play.google.com/console/about/vitals/
https://play.google.com/store/apps/details?id=com.activision.callofduty.shooter
https://play.google.com/store/apps/details?id=com.aspiro.tidal
https://play.google.com/store/apps/details?id=com.att.callprotect
https://play.google.com/store/apps/details?id=com.bethsoft.DOOM
https://play.google.com/store/apps/details?id=com.budgestudios.googleplay.ThomasAndFriendsMagicalTracks
https://play.google.com/store/apps/details?id=ch.threema.app&hl=en_US&gl=US
https://play.google.com/store/apps/details?id=com.downdogapp
https://play.google.com/store/apps/details?id=com.dragonscapes.global
https://play.google.com/store/apps/details?id=com.facebook.katana
https://play.google.com/store/apps/details?id=fi.reworks.redecor
https://play.google.com/store/apps/details?id=com.footlocker.approved
https://play.google.com/store/apps/details?id=com.infonow.bofa
https://play.google.com/store/apps/details?id=com.illumix.fnafar
https://play.google.com/store/apps/details?id=com.kii.safe
https://play.google.com/store/apps/details?id=com.king.candycrushsaga
https://play.google.com/store/apps/details?id=gov.loc.nls.dtb
https://play.google.com/store/apps/details?id=com.maize.digitalClock
https://play.google.com/store/apps/details?id=com.marvel.unlimited
https://play.google.com/store/apps/details?id=com.mojang.minecraftpe
https://play.google.com/store/apps/details?id=com.nike.ntc

https://play.google.com/store/apps/details?id=com.nike.omega

https://play.google.com/store/apps/details?id=com.nike.snkrs

https://play.google.com/store/apps/details?id=com.pacific.wildlands

https://play.google.com/store/apps/details?id=com.phone.launcher.android

https://play.google.com/store/apps/details?id=com.pixelberrystudios.choices

https://play.google.com/store/apps/details?id=com.PixelStudio

https://play.google.com/store/apps/details?id=com.thehomedepot

https://play.google.com/store/apps/details?id=com.nytimes.android

https://play.google.com/store/apps/details?id=com.zynga.livepoker

https://play.google.com/store/apps/details?id=fi.reworks.redecor

https://play.google.com/store/apps/details?id=me.lyft.android

https://play.google.com/store/apps/details?id=net.bookedin.bam

https://play.google.com/store/apps/details?id=org.kustom.wallpaper.pro

https://play.google.com/store/apps/details?id=com.raed.sketchbook

https://play.google.com/store/apps/details?id=sts.al

https://play.google.com/store/apps/developer?id=Lowtech+Studios

https://play.google.com/store/apps/details?id=com.sega.sprint

https://play.google.com/store/apps/details?id=com.tayasui.sketches

https://play.google.com/store/apps/details?id=com.symantec.mobilesecurity&hl=en_US

https://play.google.com/store/apps/details?id=com.lookout&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.robokiller.app&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.nordvpn.android&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.symantec.securewifi&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.callpod.android_apps.keeper&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.antivirus&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=hotspotshield.android.vpn&hl=en_US&gl=US

https://play.google.com/store/apps/details?id=com.avast.android.mobilesecurity&hl=en_US&gl=US

https://publishers.apps.bemobi.com//Opera-Microsoft-FAQ.html

https://seller.samsungapps.com/notice/getNoticeDetail.as?csNoticeID=0000002677

https://seller.samsungapps.com/notice/getNoticeDetail.as?csNoticeID=0000004278

https://slack.com/help/requests/new

https://steamcommunity.com/groups/steamworks/announcements/detail/1697191267930157838

https://store.wsj.com/shop/us/us/wsjuelnsb20/

https://support.google.com/googleplay/android-developer/answer/10532353

https://support.google.com/googleplay/android-developer/answer/10632485

https://support.google.com/googleplay/android-developer/answer/6112435

https://support.google.com/googleplay/android-developer/answer/9306917

https://support.google.com/googleplay/android-developer/answer/9859673

https://support.google.com/googleplay/answer/2651410

https://support.google.com/paymentscenter/answer/9003744

https://support.tidal.com/hc/en-us/articles/115003662825-Subscription-Types

https://techcrunch.com/2018/12/06/epic-games-store/

https://tinder.com/en-GB/feature/subscription-tiers

https://docs.unity.com/ads/UnityAdsHome.html

https://ubisoftconnect.com/en-US/

https://us.shop.battle.net/en-us

https://www.amazon.com/gp/help/customer/display.html?nodeId=GP96AU3MQ58FMV8U

https://www.androidauthority.com/android-market-google-play-history-754989/

https://www.androidauthority.com/best-sideloaded-apps-android-1155580/

https://www.businessofapps.com/ads/leadbolt/

https://www.centurygames.com/about/

https://www.downdogapp.com/faq

https://www.downdogapp.com/purchase

https://www.epicgames.com/fortnite/en-US/faq

https://www.epicgames.com/fortnite/en-US/mobile/android/get-started

https://www.epicgames.com/fortnite/en-US/vbuckscard

https://www.epicgames.com/site/en-US/epic-games-store-faq

https://www.epicgames.com/store/en-US/news/the-epic-games-store-is-now-live

https://www.facebook.com/audiencenetwork/

https://www.ftc.gov/system/files/documents/reports/privacy-data-security-update-2019/2019-privacy-data-security-report-508.pdf

https://www.hbomax.com/ways-to-get

https://www.help.tinder.com/hc/en-us/articles/115003356706-How-do-I-create-a-Tinder-account-

https://www.iheart.com/offers/

https://www.investopedia.com/articles/investing/102615/story-instagram-rise-1-photo0sharing-app.asp

https://www.marvel.com/unlimited

https://www.microsoft.com/en-us/store/apps/windows

https://www.minecraft.net/en-us/store/minecraft-android

https://www.mopub.com/en

https://www.myfitnesspal.com/premium?source=menu_bar

https://www.nytimes.com/2015/06/03/technology/instagram-to-announce-plans-to-expand-advertising.html

https://www.origin.com/usa/en-us/store

https://www.pandora.com/plans

https://www.paypal.com/us/business/pricing

https://www.paypalobjects.com/marketing/ua/pdf/US/en/feepages-080221.pdf

https://www.braintreepayments.com/braintree-pricing

https://stripe.com/pricing

https://squareup.com/us/en/payments/our-fees

https://www.polygon.com/2019/4/5/18295833/epic-games-store-controversy-explained

https://www.prnewswire.com/news-releases/customers-sitting-on-100-billion-of-unredeemed-loyalty-points-623828294.html

https://www.protocol.com/bulletins/microsoft-store-commission-cut

https://www.sprint.com/companyinfo/history/timeline_02.html

https://www.storemaven.com/academy/how-to-choose-an-app-category/

https://www.theverge.com/2018/11/30/18120577/valve

https://www.voodoo.io/publishing

https://www.wikiwand.com/en/Opera_Mobile_Store

https://www.federalreserve.gov/releases/h10/hist

https://www.xbox.com/en-US/microsoft-store

https://www.disneyplus.com/welcome/stream-now

https://www.microsoft.com/en-us/microsoft-365/onedrive/compare-onedrive-plans

https://myscholly.com/faqs/

https://play.google.com/store/apps/details?id=idle.empire.tycoon.bicoin

https://play.google.com/store/apps/details?id=com.taptapfun.idlebusiness

https://play.google.com/store/apps/details?id=com.gpk.CavemenIdle

https://play.google.com/store/apps/details?id=azon.chips

https://play.google.com/store/apps/details?id=net.lionbird.google.toycity

https://play.google.com/store/apps/details?id=dovi.coalmining.inc

https://play.google.com/store/apps/details?id=de.rki.covpass.app

https://play.google.com/store/apps/details?id=com.bluetick.deepsea

https://play.google.com/store/apps/details?id=com.MadDiamond.EcoRobotics

https://play.google.com/store/apps/details?id=com.kwgames.packagefactory

https://play.google.com/store/apps/details?id=com.HyperMode.GarageLand

https://play.google.com/store/apps/details?id=com.h8games.handicraft

https://play.google.com/store/apps/details?id=com.samsung.android.service.health

https://play.google.com/store/apps/details?id=com.google.android.wearable.healthservices

https://play.google.com/store/apps/details?id=com.aliensidea.houseagent

https://play.google.com/store/apps/details?id=com.adc.tty

https://play.google.com/store/apps/details?id=games.electricmonkeys.idlecompost

https://play.google.com/store/apps/details?id=com.puzzlecorp.daycare.idle.game

https://play.google.com/store/apps/details?id=com.tycoonidledentalclinic.empire

https://play.google.com/store/apps/details?id=com.jk.and.bar

https://play.google.com/store/apps/details?id=com.hwqgrhhjfd.idlefastfood

https://play.google.com/store/apps/details?id=com.neonplay.casualgolfclubtycoon

https://play.google.com/store/apps/details?id=jp.geeplus.IdleHeroHacker

https://play.google.com/store/apps/details?id=com.survival.builders

https://play.google.com/store/apps/details?id=com.forgegames.idlelaundry

https://play.google.com/store/apps/details?id=com.tbegames.and.best_idle_taps_builder_tycoon

https://play.google.com/store/apps/details?id=com.platypus.games.idle.mosquitoes

https://play.google.com/store/apps/details?id=com.holoboo.idlerescue

https://play.google.com/store/apps/details?id=com.funcell.idleschool

https://play.google.com/store/apps/details?id=com.zrl.shootingparty

https://play.google.com/store/apps/details?id=com.ludigames.android.anmp.idle.siege

https://play.google.com/store/apps/details?id=com.fanshi.zhengjing.skipark

https://play.google.com/store/apps/details?id=com.heatherglade.idlestartupper

https://play.google.com/store/apps/details?id=com.mm.idlemutou

https://play.google.com/store/apps/details?id=com.lehuo.school.an.gp

https://play.google.com/store/apps/details?id=com.montaponta.idleplanecrashsurvival

https://play.google.com/store/apps/details?id=com.Nostrovia.Jonk

https://play.google.com/store/apps/details?id=com.codigames.idle.law.empire.tycoon

https://play.google.com/store/apps/details?id=com.sablostudio.matches.craft.idle.game

https://play.google.com/store/apps/details?id=com.sablostudio.mining.tycoon.idle.game

https://play.google.com/store/apps/details?id=com.gpk.modelagency

https://play.google.com/store/apps/details?id=com.danybons.tavernmaster

https://play.google.com/store/apps/details?id=com.police.tycoon

https://play.google.com/store/apps/details?id=com.adc.iaraft

https://play.google.com/store/apps/details?id=com.warrior.ranchland.gp

https://play.google.com/store/apps/details?id=com.samsung.android.wear.shealth

https://play.google.com/store/apps/details?id=com.gpk.seaexplorers

https://play.google.com/store/apps/details?id=com.google.android.apps.security.securityhub

https://play.google.com/store/apps/details?id=com.somniumgames.roveridle

https://play.google.com/store/apps/details?id=com.metamoki.sushi

https://play.google.com/store/apps/details?id=soft.eac.startup

https://play.google.com/store/apps/details?id=cc.LionStudios.IdleFactory

https://play.google.com/store/apps/details?id=com.levelex.pantheon

https://play.google.com/store/apps/details?id=com.funcell.treesinc

https://play.google.com/store/apps/details?id=com.hg.chuanqi.usedcar

https://play.google.com/store/apps/details?id=com.soul.zhongjianshang

https://play.google.com/store/apps/details?id=com.google.android.apps.camera.services

https://play.google.com/store/apps/details?id=com.google.android.videos

https://play.google.com/store/apps/details?id=com.privateinternetaccess.android

https://play.google.com/store/apps/details?id=com.cjin.pokegenie.standard

https://play.google.com/store/apps/details?id=com.funanduseful.earlybirdalarm

https://play.google.com/store/apps/details?id=com.asmodeedigital.scythe

https://play.google.com/store/apps/details?id=com.nianticlabs.pokemongo

https://www.alltrails.com/pro?ref=header

APPENDIX C

DEFINITION OF "U.S." DEVELOPER

1. Developer Plaintiffs describe their proposed class as:

   > "All U.S. persons or entities that paid Google a "service fee" of greater than 15% on any paid Android OS app or paid in-app content (including subscriptions) sold in or via the Google Play store, in or via any U.S. or foreign Google Play storefront.[430]

2. Developer Plaintiffs have not defined what constitutes a "U.S." developer as it pertains to their proposed class definition and how such developers can be identified. Dr. Williams uses Google's App-level data to identify putative developer class members.

3. Using a Google dataset is not an appropriate way to identify "U.S." developers because Google datasets are at times not consistent. For example, App-level spend data and App Catalog data both include developers' country identifiers. However, a comparison of developers' country identifiers in the App-level spend data with the location information in the App Catalog data[431] indicates that out of the 49,387 "U.S." developers in the App-level spend data, there are 3,632 (or about 7%) that, in the App Catalog data, are identified with a location as some country other than the U.S.[432]

4. Conversely, there are developers identified as U.S. developers in Google's App Catalog data that are not identified as U.S. developers in the App-level spend data.[433] Because the App Catalog identifies the country of each app, in some cases a developer identified in the App-level spend data as a U.S. developer may have apps identified as U.S. apps, non-U.S. apps, or a combination of U.S. and non-U.S. apps. Among the 49,387 "U.S." developers in the App-level spend data, 81% are identified as U.S. developers in the App Catalog data, and 19% are either identified as non-U.S. developers or have no country information in the App Catalog data. Exhibit 62 shows these results.

5. [redacted]

---

430 Developer Complaint at ¶244.

431 The App-level spend data and App Catalog data do not include the same unique developer identifiers. To find matching developers in the two datasets, "app package names" are matched. A developer associated with a unique set of app package names in App-level spend data is "matched" to the developer associated with the same unique set of app package names in the App Catalog data. The App Catalog data is a "snapshot" of data as of May 5, 2021. Therefore, the App Catalog data does not show changes in app ownership over time and for that reason, does not provide precise matches to developers in the App-level spend data.

432 See Exhibit 62. 5,836 developers, or 12%, have no country information available in the App Catalog data.

433 See this report's production.

434 https://www.centurygames.com/about/

[redacted][435]

6. On the other hand, [redacted] is identified in the App-level spend data with the country code of [redacted]. Yet [redacted] has been incorporated in the U.S. since 1982 – first in California and later in Delaware – and its principal executive offices are in California.[436]

7. Whether some of the developers considered "U.S." developers by Dr. Williams and Developer Plaintiffs are, in fact, U.S. developers depends on the definition of U.S. developer – which was not provided in the proposed class definition.

8. Century Games and Electronic Arts are only examples. Whether one or the other should be considered a U.S. developer could depend on criteria that must be collected and assessed on an individualized basis.

9. There are additional issues about developers' class definition that must be addressed. In the App-level spend data, subsidiary companies of the same parent are identified as separate developers. In some cases, a company and its subsidiary may have different country locations. For purposes of this report, unless otherwise described, a developer identified in the App-level spend data is considered a unique developer and no attempt has been made to account for cross-ownership among the developers. Nonetheless, the corporate structure is likely important to the identification of at least some prospective members of the putative developer class.

10. There also is further ambiguity in the class definition regarding the term "developer." A developer could be the *creator* of the app or the *publisher* of the app – which, in some cases, are different entities.[437] If the class definition is intended to refer to *creator*-developers, then some proposed class members would not appear in Google data at all, since only the publisher-developers interact with Google. Moreover, the terms of the relationship between the *publisher*-developer and the *creator*-developer are not available in Google data (or generally known by Google) and could imply that reducing the service fee charged to the *publisher*-developer would not affect the *creator*-developer. If the class definition is intended to include *creator*-developers, individualized analysis would be required not only to identify the proposed class member but to determine whether the proposed class member could have been impacted by an allegedly high service fee rate. The Developer Plaintiffs' class definition refers to "U.S. persons or entities who paid Google a service fee" which

---

435 Century Games Pte Limited, Century Games Limited, and Century Games Publishing are developers in the App-level Data used by Dr. Williams with a country code of "U.S." See this report's production.

436 Electronic Arts Inc. Form 10-K, 2020, at p. 6

437 See, for example, the developer [redacted], who is an independent creator-developer versus a publisher-developer such as [redacted] who contracts with creators. Both [redacted] and [redacted] are identified in Google Play as developers while the developer-creators who contract with [redacted] are not identified in Google Play or in Google Play data. See https://play.google.com/store/apps/details?id=com.maize.digitalClock; https://ezi.am/; https://www.voodoo.io/publishing.

would suggest a publisher. But if a publisher-developer passes the fee onto a creator-developer, it is unclear how these entities fit into the Plaintiffs' definition.

APPENDIX D

SUMMARY OF DEVELOPER TOOLS AND SERVICES IN GOOGLE PLAY

1. This appendix provides a brief summary of services and tools made available to developer that use Google Play based on publicly available information and evidence from the record in this case. It is not intended to be a comprehensive description of all services available in Google Play or to catalogue all facts about those services relevant to the merits of Plaintiffs' [illegible]

2. The services available to all Google Play developers include four categories of services: developer publishing tools; developer commerce tools; app discovery; and app distribution. These tools are available to all developers, and they are especially valuable to small developers who may not have these same types of resources available to them from other sources.

3. Google Play provides many tools that enable developers to publish apps for consumers who use the diverse set of Android mobile devices. There are tens of thousands of different combinations of Android device characteristics relevant to the use of apps – characteristics such as the amount of memory, the memory configuration of the device, the CPU, the GPU, screen resolution, and others.[438] Whether an app will work at all, or will work well, can differ across devices and depend on those configurations. Google Play offers tools that enable developers to improve app functionality across the diverse set of Android mobile devices and to assist in the management and marketing of apps. These include tools used by developers prior to the release of the app to test the app by a small group of trusted users or by a larger group,[439] to reduce the size of the app to save storage space on a consumer device and reduce latency that may lead to lower spend on an app,[440] and to access and incorporate app bundles that contain the elements an app needs to install correctly on mobile devices.[441]

---

438 Lim Dep. at pp. 59, 228.

439 https://play.google.com/console/about/closed-testing/; https://play.google.com/console/about/internal-testing/; https://play.google.com/console/about/opentesting/

440 https://play.google.com/console/about/app-bundle-explorer/

441 https://play.google.com/console/about/internalappsharing. See also Lim Dep. at p. 63. Google offers many other publishing tools to developers. See e.g., https://play.google.com/console/about/devicecatalog/ ("device catalog" that can be used to view and identify mobile devices that are compatible with the app); https://play.google.com/console/about/keymanagement/ (security "keys" for publishing the app); https://play.google.com/console/about/pre-registration/ (pre-registration which allows users to receive a notification when the app is available or to have the app auto-installed at launch); https://play.google.com/console/about/storelistings/ (help in creating store listings that appeal to users); https://play.google.com/console/about/store-listing-experiments/ (ways for developers to experiment and test alternative listings); https://play.google.com/console/about/translationservices/ (Google Play allows apps to be translated into many languages and offers automated translations service); https://play.google.com/console/about/acquisitionreporting/; https://play.google.com/console/about/stats/ (services that allow developers to track and analyze user acquisition trends);

4. Google Play provides developers with the ability to engage in a myriad of different types of paid transactions. Through Google Play Billing, developers are able to efficiently receive payments from an international audience.[442] Google Play facilitates transactions in over 135 countries, applying the relevant taxes across numerous different jurisdictions in varying different currencies.[443] Google Play also provides the means to transact with various different forms of payment, including credit cards, debit cards, PayPal, DCB, Google Pay, and gift cards used in the countries in which the store is available.[444]

5. The availability of payment processing services across many countries is important to those developers who offer apps available internationally. Among the 49,387 "U.S." developers, 75% have some sales outside of the U.S. and 37% earn more than 50% of their revenue from consumers outside of the U.S.[445]

6. Google's costs of processing transactions vary across transaction type. The cost of processing credit card and PayPal transactions is [redacted], but the cost of processing gift cards is [redacted] and the average cost of DCB is [redacted].[446] Since different developers have varying mixes of transaction types, Google's cost of processing transactions varies across developers,

---

https://play.google.com/console/about/playgamesservices/ (Play Game Services provides game app developers with features such as automatic sign-in, leaderboards and others); https://play.google.com/console/about/vitals/ (Android Vital is a service that allows developers to monitor and improve the technical quality of apps by providing information to prioritize and debug issues impacting users.).

[442] https://developer.android.com/distribute/play-billing ("Google Play supports over 135 countries to buy and sell digital goods, which means you can sell to a global audience without collecting separate payment information or looking into regulations in each country."). See also https://support.google.com/googleplay/android-developer/answer/10532353 (publicly available list of. 172 "consumer locations" for which Google supports paid transactions).

[443] See https://support.google.com/googleplay/android-developer/answer/9306917 (showing varying currencies supported by Google Play).

[444] https://developer.android.com/distribute/google-play; https://support.google.com/googleplay/answer/2651410

[445] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689. See this report's production. Through Google Play's billing system, Google processes and pays the fees on behalf of developers and manages a payment infrastructure that makes it easy and safe for consumers to engage in transactions. Google Play's fraud protection encourages consumers to engage in transactions. Google offers chargeback protection to developers; it will evaluate chargebacks and fight them on the developer's behalf. In addition, Google has automatic fraud detection by using advanced risk modeling and leverages cross-industry resources like the worldwide fraud blocklist. They will cancel any active orders associated with the same fraudulent credit card. https://support.google.com/paymentscenter/answer/9003744.

[446] See GOOG-PLAY-000337564 at 587. These costs represent Google's cost; it may be difficult for developers to replicate the processing costs and therefore developers who process their own payments could have higher costs than Google's.

and the costs incurred by developers who elected to process their own payments would also vary depending on the mix of payment types.[447]

7. App discovery refers to the way in which consumers become aware of an app. In general, developers obtain discovery through app stores, such as Google Play, or by purchasing advertising. Google estimated that in 2018, over 76 billion app installs and 32 billion app reinstalls were generated through Google Play features that aided consumer discovery of apps.[448] Google Play services are especially important to developers who do not have a recognized reputation or brand or who do not have the resources necessary to invest in advertising and marketing themselves to promote their apps. For some developers, Google Play discovery features are a more effective way to obtain consumer installs compared to the cost of obtaining installs through advertising.[449]

8. Another important service provided by Google Play is the delivery of the app, and its updates, to the consumer's device.[450] Google Play delivery is designed to ensure that the app is genuine and not malicious, and that the app is not corrupted or damaged through delivery. For example, if the installation is interrupted due to loss of data, Google Play will pause the installation and resume when data becomes available.[451] Google Play Protect is a security feature to ensure apps are not malicious. Each day Google Play Protect scans all of the apps on Android phones and works to prevent the installation of harmful apps and keep the devices and data safe.[452]

9. Delivery costs vary across apps depending on the size of the app, the number of consumers who have installed the app, and the frequency of updates. Thus, changes in app delivery may leave some developers – those most dependent on current services – worse off.

---

447 See GOOG-PLAY-000337564 at 630 (showing varying percentages of payment types across different categories of developers).

448 See GOOG-PLAY-000541836. There are two other categories identified in the document: Deep Links, which are links to the app created by the developer and Other, which are not described in the document.

449 Google Play also provides the opportunity for consumers to pre-register for an app prior to its launch and receive notification when it is available. See https://developer.android.com/distribute/best-practices/launch/pre-registration. Google Play Instant and the Try Now feature allows consumers to try an app without having to install or pay for it. See https://developer.android.com/topic/google-play-instant/overview; https://developer.android.com/distribute/google-play. Through Google Play Instant, developers publish an "instant app" that allows consumers to see a "demo" version or a "lite" version of the app. This feature encourages consumers to try apps, which ultimately leads to increased installations and revenue generation for developers. https://developer.android.com/topic/google-play-instant/overview.

450 In its business documents, Google refers to the delivery of the app and its updates as "distribution."

451 See GOOG-PLAY-000286913 at 917.

452 https://developers.google.com/android/play-protect/phacategories; https://developers.google.com/android/play-protect

10. Google also provides customized services to targeted developers. Project Hug offers [redacted].[453] In 2019, Google planned to spend over [redacted] on [redacted] developers and estimated that the value of these programs to the developers were worth over [redacted].[454]

---

[453] See GOOG-PLAY-000565172.R at slides 18-19.

[454] See GOOG-PLAY-000237766. Note that Hug developers are also considered Priority Partner Program developers and Alliance developers. See GOOG-PLAY-000521845. Google offers other developers varying levels of customized support and solutions. Google's Priority Partner Program ("PPP"), Alliance developers, Core developers, Majors developers, and Ecosystem developers are categories of developers whom Google targets and provides services to induce the developers to build or maintain relationships with Google Play. Alliance and Major developers are considered the "most critical" developers to Google Play. These developers obtain customized, or "high touch" marketing and technical consultation from Google, as well as other incentives. See GOOG-PLAY-000232540 at slides 30, 34.

Exhibit 1

Number of Free and Paid/Subscription/IAP Apps in Google Play

Active Apps Listed by Any Developer on May 5, 2021

	Paid/Subscription/IAP Apps		Free Apps		All Apps	
	Count	Percentage	Count	Percentage	Count	Percentage
Non-Game Apps	[redacted]					
Game Apps						
Total						

Apps Listed by Putative Developer Class Members, August 2016 - December 2021

	Paid/Subscription/IAP Apps		Free Apps		All Apps	
	Count	Percentage	Count	Percentage	Count	Percentage
Non-Game Apps	[redacted]					
Game Apps						
Total						

Notes:

[1] In the top panel, "Paid/Subscription/IAP apps" and "Free Apps" are determined from how the Play App Catalog lists the app. In the bottom panel, "Paid/Subscription/IAP apps" are apps that had sales according to the app-level spend data. "Free Apps" include completely free apps and free apps with advertising.

[2] Apps are included in the bottom panel only if they appear in any of the four sources between August 2016 and December 2021.

[3] This top panel is based on Android apps in Google Play on May 5, 2021. It omits Android apps not registered in Google Play and the breakdown could be different for Apple iOS apps or Microsoft Windows apps.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

[3] Global Google Admob Data, GOOG-PLAY-009908838

[4] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 2

Number and Percent of Non-Game Apps that are Free, by Category

Published Apps as of May 5, 2021

App Category	Number of Free Apps	Total Number of Apps	Percent of Apps that are Free
Education	359,156	400,091	90%
Business	306,157	313,756	98%
Tools	240,341	268,208	90%
Music & Audio	242,974	253,109	96%
Entertainment	225,678	244,563	92%
Lifestyle	176,847	189,543	93%
Food & Drink	181,857	185,315	98%
Shopping	181,168	183,074	99%
Productivity	162,096	175,171	93%
Books & Reference	150,059	171,172	88%
Health & Fitness	131,016	143,879	91%
Personalization	106,530	128,598	83%
Finance	116,763	122,555	95%
Travel & Local	100,256	109,492	92%
Communication	91,872	96,794	95%
Social	79,261	86,183	92%
News & Magazines	60,091	66,765	90%
Sports	57,907	66,434	87%
Medical	55,244	60,954	91%
Maps & Navigation	46,975	50,733	93%
Photography	45,140	49,376	91%
Auto & Vehicles	41,256	43,501	95%
House & Home	28,404	29,286	97%
Art & Design	26,773	29,049	92%
Beauty	25,655	26,195	98%
Events	24,544	25,030	98%
Video Players	18,752	21,014	89%
Dating	8,522	11,834	72%
Libraries & Demo	10,039	10,789	93%
Weather	7,771	10,069	77%
Parenting	5,638	6,847	82%
Comics	5,110	6,041	85%
Transportation	44	46	96%
Media & Video	23	24	96%
Travel	2	2	100%
Total Non-Game	**3,319,921**	**3,585,492**	**93%**

Notes:

[1] The table includes only active apps. "Free apps" have no paid downloads, no IAPs, and no subscriptions.

[2] "Free apps" may include apps with advertising.

[3] Percentages are with respect to the total count of apps in each category.

[4] Apps with unknown categories are excluded from analysis.

Source:

[1] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 3

Number and Percent of Game Apps that are Free, by Category

Published Apps as of May 5, 2021

App Category	Number of Free Apps	Total Number of Apps	Percent of Apps that are Free
Arcade	66,403	84,198	79%
Casual	63,865	81,400	78%
Puzzle	57,968	78,987	73%
Action	29,608	42,179	70%
Educational	29,252	38,306	76%
Adventure	26,768	35,845	75%
Simulation	21,083	31,843	66%
Trivia	16,874	20,909	81%
Board	12,091	16,316	74%
Role Playing	7,131	15,978	45%
Word	9,836	14,539	68%
Racing	10,121	14,415	70%
Strategy	8,635	14,403	60%
Card	8,108	12,820	63%
Sports	8,550	12,157	70%
Casino	5,153	9,224	56%
Music	4,807	6,078	79%
Total Game	**386,253**	**529,597**	**73%**

Notes:

[1] The table includes only active apps. "Free apps" have no paid downloads, no IAPs, and no subscriptions.
[2] "Free apps" may include apps with advertising.
[3] Percentages are with respect to the total count of apps in each category.
[4] Apps with unknown categories are excluded from analysis.

Source:

[1] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 4

Percent of Apps and Consumer Spend with Subscriptions or IAPs

August 2016 - December 2021

	Percentage of Apps With Subscriptions or IAPs	Subscriptions and IAPs as a Percentage of Consumer Spend	Subscriptions and IAPs as a Percentage of Google Play Service Fees
Proposed Developer Class Members	43%	99%	99%
Proposed Consumer Class Members	54%	99%	N/A

Sources:

[1] App-level Spend Data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] App-level Spend Data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

Exhibit 5

Range of Promotion Amounts in Play Store Transactions

August 2016 - July 3, 2021

	Min Amount ($)	Max Amount ($)	Min % Discount	Max % Discount
All Promos (Pre-Tax)	$0.01	$399.99	0.11%	100%
Developer Promos	$0.10	$399.99	0.95%	100%
Google Promos	$0.01	$239.00	0.11%	100%
List Prices	$0.05	$399.99		

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] List Price is price_per_unit_pre_tax_micros and Net Price is sale_revenue_pre_tax_micros, adjusted for quantity. Developer promotion is the amount from Partner (Developer) to Buyer. Google promotion is amount from Google to Buyer.

[3] The promo amount is defined as (List Price - Net Price).

[4] Developer and Google promotion amounts are observed separately in the data only on a post-tax basis.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 6

Promotions by Number of Months with Transactions

August 2016 - July 3, 2021

	Count of Unique Promotions	Cumulative Percent
1 Month or less	644,912	60%
2 to 5 Months	305,919	89%
6 to 12 Months	76,793	96%
13 to 24 Months	29,593	99%
25 to 36 Months	8,511	99.8%
More than 36 Months	1,715	100%
Total	**1,067,443**	**100%**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Unique promotions are defined as a unique combination of product ID, list price, net price, and discount amount.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 7

Developers with Free or Paid/Subscription/IAP Apps in Google Play App Catalog

Snapshot as of May 5, 2021

Developers with:	Count of Developers	Percent of Developers
Only Free Apps	1,191,800	87%
Free and Paid/Subscription/IAP Apps	75,495	6%
Only Paid/Subscription/IAP Apps	94,844	7%
Total	**1,362,139**	**100%**

Notes:

[1] "Free Apps" may include apps with advertising.

[2] Developer counts based on the Play Catalog's developer account IDs associated with active apps.

Source:

[1] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 8

Deal Developers by Program

2021

Program	Putative Developer Class Members	Count	Other Developers	Count
LRAP	[redacted]	25	[redacted]	20
LRAP++	[redacted]	1	[redacted]	1
SwG	[redacted]	18	[redacted]	81
ADAP	[redacted]	2	[redacted]	3
Project Hug/GVP	[redacted]	6	[redacted]	14

Notes:

[1] A deal developer is defined as being in the putative class if it appears in the App-level spend data, Developers.
[2] [redacted] and [redacted] are in both SwG and LRAP.
[3] [redacted], and [redacted] have foreign entities as well.
[4] "U.S." and foreign entities are listed for developer with both.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689
[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688
[3] GOOGLE-PLAY-001291192; GOOG-PLAY-000237766; Marchak Deposition Exhibit 370
[4] See this report's production for Bates numbers of deal contracts and app package names.

Exhibit 9

Putative Developer Class Members By Consumer Spend

August 2016 - December 2021

Global Consumer Spend	Number of Developers	Percent	Cumulative Percent
$0 or less	449	1%	1%
>$0 to $100	26,486	54%	55%
>$100 to $1,000	9,356	19%	73%
>$1,000 to $10,000	6,916	14%	87%
>$10,000 to $100,000	3,745	8%	95%
>$100,000 to $1,000,000	1,672	3%	98%
>$1,000,000 to $10,000,000	546	1%	99.6%
>$10,000,000 to $100,000,000	170	0.3%	99.9%
>$100,000,000	47	0.1%	100%
Total	**49,387**	**100%**	**100%**

Notes:

[1] Consumer spend includes all sales from paid downloads, subscriptions, and IAPs.

[2] Apps may have net negative consumer spend due to refunds or promotions.

Source:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 10

Comparison of Putative Developer Class Members, Dr. Williams and Dr. Burtis

	Dr. Williams	Dr. Burtis
Class Period	[redacted]	
Number of Developers		
Developers' Final Consumer Spend		
Final Google Revenue		

Note:
[1] The collapsed cleaned versions of the data from each respective expert is used.

Sources:
[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689
[2] Williams backup

Exhibit 11

Putative Developer Class Members by App Characteristics, Monetization Type and Consumer Spend

	Aug 2016 - Dec 2021		2016 (Aug - Dec)		2017		2018		2019		2020		2021	
	Count	Percent	Count	Percent	Count	Percent	Count	Percent	Count	Percent	Count	Percent	Count	Percent
Number of Paid/Subscription/IAP Apps	132,593		57,345		77,373		68,210		50,185		46,772		47,700	
Number of Putative Developer Class Members	49,387	100%	19,863	100%	25,669	100%	25,174	100%	19,621	100%	19,995	100%	21,687	100%
Putative Developer Class Members Using:														
In-App Purchases	19,695	40%	6,317	32%	8,424	33%	8,510	34%	8,264	42%	8,310	42%	8,656	40%
Paid Downloads	26,490	54%	14,078	71%	17,695	69%	16,284	65%	9,856	50%	8,999	45%	8,212	38%
Subscriptions	10,820	22%	1,757	9%	2,695	10%	3,557	14%	4,505	23%	5,824	29%	8,077	37%
Both Paid Downloads and IAPs/Subscriptions	3,998	8%	1,621	8%	2,104	8%	1,915	8%	1,554	8%	1,380	7%	1,236	6%
Putative Developer Class Members with Sales of:														
Non-Game Apps Only	33,140	67%	14,453	73%	18,245	71%	18,160	72%	13,829	70%	14,347	72%	15,658	72%
Game Apps Only	14,083	29%	4,665	23%	6,272	24%	6,075	24%	5,175	26%	5,120	26%	5,493	25%
Both Non-Game & Game Apps	2,164	4%	745	4%	1,152	4%	939	4%	617	3%	528	3%	536	2%
Putative Developer Class Members with Annual Spend < $99	28,039	57%	11,208	56%	16,591	65%	15,980	63%	10,315	53%	10,304	52%	10,194	47%
Putative Developer Class Members with Annual Service Fees < $99	33,334	67%	13,764	69%	19,178	75%	18,540	74%	12,615	64%	12,597	63%	12,904	60%

Notes:

[1] Total developer counts are less than the sum of each individual year because developers may have multiple monetization types or sales transactions over multiple years.

[2] In Aug 2016 - Dec 2021, the number of putative class developers with consumer spend or service fees of less than $99 includes those developers who earned less than $99 in every year they had sales transactions over the class period.

[3] Number of putative class developers with consumer spend or service fees of less than $99 calculates annual spend or service fees for 2016 but only includes developers with sales in the class period, August 2016 - December 2021.

[4] When a developer puts an app on Google Play they have to pay a $25 fee. This fee is seen as revenue during their first year. The first year is determined by the Catalog Data or the App-level spend data, whichever is seen first.

Sources:

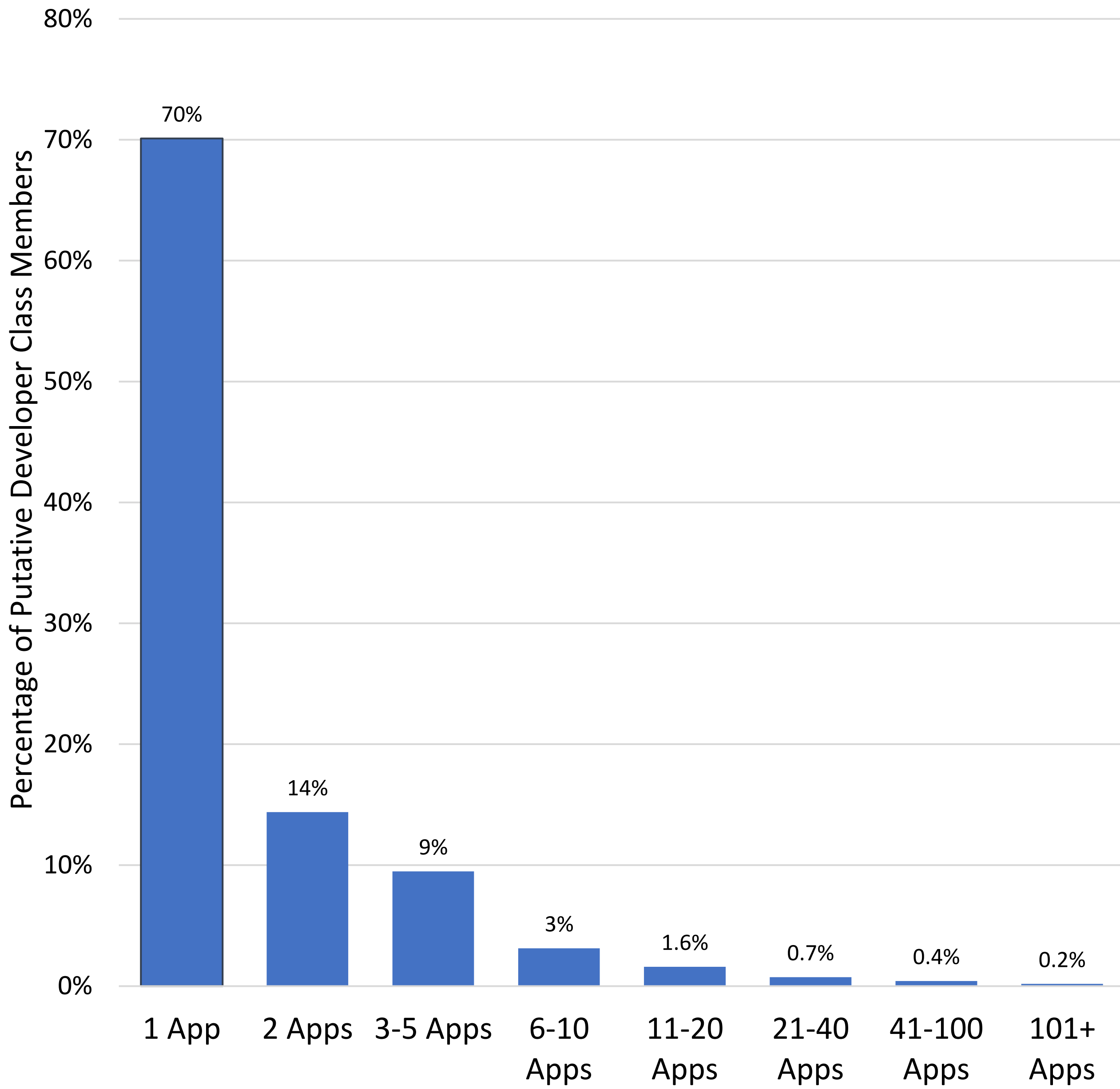
[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 12

Percentage of Putative Developer Class Members Offering Different Numbers of Paid/Subscription/IAP Apps

August 2016 - December 2021



Source: App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 13

[redacted]

Exhibit 14

Top 30 Putative Developer Class Members Accounting for 75% of Consumer Spend

2021 by Developer Parent Groups ($ in millions)

Developer	Rank	Consumer Spend	Percent of Spend	Cumulative Percent of Spend
[redacted]	[redacted]			
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]n				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				
[redacted]				

Notes:

[1] Parent - Subsidiary relationship is as of December 2021.

[2] Putative class parent groups only include the putative class subsidiary developers of the parent group.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

[3] Public internet sources, May 25th, 2021 - March 3rd, 2022

Exhibit 15

Putative Developer Class Members Offering Apps with Ads

August 2016 - December 2021

	Count of Developers	% of Developers
Developer has NO apps with Ads	16,733	34%
Developer has at least one app with Ads	32,654	66%
All Developers	49,387	100%

Notes:

[1] A developer has advertising if he has an app identified in the App Catalog as including advertising.

[2] If the developer does not have an app in the App Catalog, it is assumed there are no ads.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 16

Example of Putative Developer Class Members Apps with Varying Effective Service Fee Rates

August 2016 – December 2021

Developer Name	App Name	Consumer Spend	AdMob Earnings	Observed Service Fee Rate	Effective Service Fee Rate
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Notes:

[1] AdMob earnings are net of revenue shares.

[2] These Android apps were found only on Google Play Store and not on other Android app stores that AdMob supports as of Dec 13, 2021.

[3] Service fee rates use pre-tax revenue and spend amounts.

[4] The app "Senior Safety Phone - Big Icons Launcher" changed its app package name around 2018. Both app packages are included.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Global Google Admob Data, GOOG-PLAY-009908838

[3] Google Play; Amazon App Store; Samsung Galaxy Store, VIVO App Store; OPPO App Store, Xiaomi GetApps, and https://support.google.com/admob/answer/10037806, accessed December 13, 2021

Exhibit 17

Relative Size of Deal Developers, 2021

Counts and spend associated with apps enrolled in deal programs

Program	Number of Putative Developer Class Members	Percentage of Putative Developer Class Members' Spend in 2021	Number of Developers	Percentage of U.S. Consumer Spend in 2021
[redacted]				

Notes:

[1] [redacted] are in both SwG and LRAP.
[2] [redacted] have non-US entities.
[3] All of the developer's apps are included for GVP/Project Hug developers. Only subscription apps in the deals are included for developers' spend in the other programs.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689
[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688
[3] GOOGLE-PLAY 001291192; Marchak Deposition Exhibit 370
[4] See this report's production for Bates numbers of deal contracts and app package names.

Exhibit 18

Top Putative Developer Class Members Accounting for 90% of Service Fees

2021 by Developer Parent Groups ($ in millions)

Developer	Rank	Service Fees	Percent of Service Fees	Cumulative Percent of Service Fees
[redacted]	1	[redacted]	[redacted]	[redacted]
[redacted]	2	[redacted]	[redacted]	[redacted]
[redacted]	3	[redacted]	[redacted]	[redacted]
[redacted]	4	[redacted]	[redacted]	[redacted]
[redacted]	5	[redacted]	[redacted]	[redacted]
[redacted]	6	[redacted]	[redacted]	[redacted]
[redacted]	7	[redacted]	[redacted]	[redacted]
[redacted]	8	[redacted]	[redacted]	[redacted]
[redacted]	9	[redacted]	[redacted]	[redacted]
[redacted]	10	[redacted]	[redacted]	[redacted]
[redacted]	11	[redacted]	[redacted]	[redacted]
[redacted]	12	[redacted]	[redacted]	[redacted]
[redacted]	13	[redacted]	[redacted]	[redacted]
[redacted]	14	[redacted]	[redacted]	[redacted]
[redacted]	15	[redacted]	[redacted]	[redacted]
[redacted]	16	[redacted]	[redacted]	[redacted]
[redacted]	17	[redacted]	[redacted]	[redacted]
[redacted]	18	[redacted]	[redacted]	[redacted]
[redacted]	19	[redacted]	[redacted]	[redacted]
[redacted]	20	[redacted]	[redacted]	[redacted]
[redacted]	21	[redacted]	[redacted]	[redacted]
[redacted]	22	[redacted]	[redacted]	[redacted]
[redacted]	23	[redacted]	[redacted]	[redacted]
[redacted]	24	[redacted]	[redacted]	[redacted]
[redacted]	25	[redacted]	[redacted]	[redacted]
[redacted]	26	[redacted]	[redacted]	[redacted]
[redacted]	27	[redacted]	[redacted]	[redacted]
[redacted]	28	[redacted]	[redacted]	[redacted]
[redacted]	29	[redacted]	[redacted]	[redacted]
[redacted]	30	[redacted]	[redacted]	[redacted]
[redacted]	31	[redacted]	[redacted]	[redacted]
[redacted]	32	[redacted]	[redacted]	[redacted]
[redacted]	33	[redacted]	[redacted]	[redacted]
[redacted]	34	[redacted]	[redacted]	[redacted]
[redacted]	35	[redacted]	[redacted]	[redacted]
[redacted]	36	[redacted]	[redacted]	[redacted]
[redacted]	37	[redacted]	[redacted]	[redacted]
[redacted]	38	[redacted]	[redacted]	[redacted]
[redacted]	39	[redacted]	[redacted]	[redacted]
[redacted]	40	[redacted]	[redacted]	[redacted]
[redacted]	41	[redacted]	[redacted]	[redacted]
[redacted]	42	[redacted]	[redacted]	[redacted]
[redacted]	43	[redacted]	[redacted]	[redacted]
[redacted]	44	[redacted]	[redacted]	[redacted]
[redacted]	45	[redacted]	[redacted]	[redacted]
[redacted]	46	[redacted]	[redacted]	[redacted]
[redacted]	47	[redacted]	[redacted]	[redacted]

Exhibit 18 (Continued)

Top Putative Developer Class Members Accounting for 90% of Service Fees

2021 by Developer Parent Groups ($ in millions)

Developer	Rank	Service Fees	Percent of Service Fees	Cumulative Percent of Service Fees
[redacted]	48	[redacted]	[redacted]	[redacted]
[redacted]	49	[redacted]	[redacted]	[redacted]
[redacted]	50	[redacted]	[redacted]	[redacted]
[redacted]	51	[redacted]	[redacted]	[redacted]
[redacted]	52	[redacted]	[redacted]	[redacted]
[redacted]	53	[redacted]	[redacted]	[redacted]
[redacted]	54	[redacted]	[redacted]	[redacted]
[redacted]	55	[redacted]	[redacted]	[redacted]
[redacted]	56	[redacted]	[redacted]	[redacted]
[redacted]	57	[redacted]	[redacted]	[redacted]
[redacted]	58	[redacted]	[redacted]	[redacted]
[redacted]	59	[redacted]	[redacted]	[redacted]
[redacted]	60	[redacted]	[redacted]	[redacted]
[redacted]	61	[redacted]	[redacted]	[redacted]
[redacted]	62	[redacted]	[redacted]	[redacted]
[redacted]	63	[redacted]	[redacted]	[redacted]
[redacted]	64	[redacted]	[redacted]	[redacted]
[redacted]	65	[redacted]	[redacted]	[redacted]
[redacted]	66	[redacted]	[redacted]	[redacted]
[redacted]	67	[redacted]	[redacted]	[redacted]
[redacted]	68	[redacted]	[redacted]	[redacted]
[redacted]	69	[redacted]	[redacted]	[redacted]
[redacted]	70	[redacted]	[redacted]	[redacted]
[redacted]	71	[redacted]	[redacted]	[redacted]
[redacted]	72	[redacted]	[redacted]	[redacted]
[redacted]	73	[redacted]	[redacted]	[redacted]
[redacted]	74	[redacted]	[redacted]	[redacted]
[redacted]	75	[redacted]	[redacted]	[redacted]
[redacted]	76	[redacted]	[redacted]	[redacted]
[redacted]	77	[redacted]	[redacted]	[redacted]
[redacted]	78	[redacted]	[redacted]	[redacted]
[redacted]	79	[redacted]	[redacted]	[redacted]
[redacted]	80	[redacted]	[redacted]	[redacted]
[redacted]	81	[redacted]	[redacted]	[redacted]
[redacted]	82	[redacted]	[redacted]	[redacted]
[redacted]	83	[redacted]	[redacted]	[redacted]
[redacted]	84	[redacted]	[redacted]	[redacted]
[redacted]	85	[redacted]	[redacted]	[redacted]
[redacted]	86	[redacted]	[redacted]	[redacted]
[redacted]	87	[redacted]	[redacted]	[redacted]
[redacted]	88	[redacted]	[redacted]	[redacted]
[redacted]	89	[redacted]	[redacted]	[redacted]
[redacted]	90	[redacted]	[redacted]	[redacted]
[redacted]	91	[redacted]	[redacted]	[redacted]
[redacted]	92	[redacted]	[redacted]	[redacted]
[redacted]	93	[redacted]	[redacted]	[redacted]
[redacted]	94	[redacted]	[redacted]	[redacted]

Exhibit 18 (Continued)

Top Putative Developer Class Members Accounting for 90% of Service Fees

2021 by Developer Parent Groups ($ in millions)

Developer	Rank	Service Fees	Percent of Service Fees	Cumulative Percent of Service Fees
[redacted]	95	[redacted]	[redacted]	[redacted]
[redacted]	96			
[redacted]	97			
[redacted]	98			
[redacted]	99			
[redacted]	100			
[redacted]	101			
[redacted]	102			
[redacted]	103			
[redacted]	104			
[redacted]	105			
All Other Putative Class Developers				
Total				

Note:

[1] Parent - subsidiary relationship is as of December 2021. Parent groups include only putative class subsidiary developers. The original developer name is shown if there is only one developer in the parent group in the 2021 app-level data.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

[3] Public internet sources, May 25th, 2021 - March 3rd, 2022

Exhibit 19

Top Putative Developer Class Members Accounting for 90% of Service Fees

August 2016 - December 2021

Developer	Rank	Consumer Spend	Service Fees	Percentage of Service Fees	Cumulative Percentage of Service Fees
[redacted]	1	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	2	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	3	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	4	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	5	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	6	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	7	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	8	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	9	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	10	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	11	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	12	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	13	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	14	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	15	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	16	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	17	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	18	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	19	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	20	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	21	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	22	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	23	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	24	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	25	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	26	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	27	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	28	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	29	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	30	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	31	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	32	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	33	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	34	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	35	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	36	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	37	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	38	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	39	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	40	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	41	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	42	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	43	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	44	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	45	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	46	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	47	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	48	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	49	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	50	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	51	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	52	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	53	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	54	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	55	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	56	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	57	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	58	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	59	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	60	$ [redacted]	[redacted]	[redacted]	[redacted]

Exhibit 19 (Continued)

Top Putative Developer Class Members Accounting for 90% of Service Fees

August 2016 - December 2021

Developer	Rank	Consumer Spend	Service Fees	Percentage of Service Fees	Cumulative Percentage of Service Fees
[redacted]	61	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	62	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	63	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	64	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	65	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	66	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	67	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	68	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	69	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	70	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	71	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	72	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	73	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	74	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	75	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	76	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	77	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	78	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	79	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	80	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	81	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	82	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	83	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	84	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	85	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	86	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	87	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]e	88	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	89	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	90	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	91	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	92	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	93	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	94	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	95	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	96	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	97	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	98	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	99	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]y	100	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	101	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	102	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	103	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	104	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	105	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	106	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	107	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	108	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	109	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	110	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	111	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	112	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	113	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	114	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	115	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	116	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	117	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	118	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	119	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	120	$ [redacted]	[redacted]	[redacted]	[redacted]

Exhibit 19 (Continued)

Top Putative Developer Class Members Accounting for 90% of Service Fees

August 2016 - December 2021

Developer	Rank	Consumer Spend	Service Fees	Percentage of Service Fees	Cumulative Percentage of Service Fees
[redacted]	121	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	122	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted].	123	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	124	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	125	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	126	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	127	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	128	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	129	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	130	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	131	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	132	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	133	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	134	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	135	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	136	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	137	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	138	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	139	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	140	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	141	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	142	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	143	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	144	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	145	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	146	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	147	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	148	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	149	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	150	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	151	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	152	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	153	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	154	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	155	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	156	$ [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	157	$ [redacted]	[redacted]	[redacted]	[redacted]
All Other Putative Class Developers		$ [redacted]	[redacted]	[redacted]	[redacted]
Total		$ [redacted]	[redacted]	[redacted]	[redacted]

Note:

[1] Consumer spend and service fees are from worldwide consumers during the class period.

Source:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 20

Number of U.S. Consumers and Number of Apps with Sales to U.S. Consumers

Year	Number of U.S. Consumers	Number of Apps with Sales to U.S. Consumers
2016	31,124,726	187,418
2017	32,689,639	197,347
2018	35,855,542	188,392
2019	38,635,466	161,517
2020	44,425,106	158,099
2021	36,619,126	136,159
Aug 2016 - Dec 2020	**83,850,236**	**378,642**
Aug 2016 - July 3, 2021	**92,329,268**	**404,325**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Unique apps are counted from apps with in-app purchases, in-app subscription, or paid download sales.

[3] Each consumer is identified in the data with a "hashed purchase initiator Google ID."

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 21

Number of Apps from which U.S. Consumers Purchase

August 2016 – July 3, 2021

Number of Apps	Number of Consumers	Percent of Consumers	Cumulative Percent of Consumers
Only 1 App	37,690,509	41%	41%
2 Apps	15,369,289	17%	57%
3 Apps	8,932,225	10%	67%
4 Apps	5,921,644	6%	74%
5 Apps	4,240,558	5%	78%
6 Apps	3,177,880	3%	82%
7 Apps	2,468,276	3%	84%
8 Apps	1,962,423	2%	86%
9 Apps	1,594,409	2%	88%
10 Apps	1,313,045	1%	90%
>10 to 50 Apps	9,101,126	10%	99%
>50 to 100 Apps	468,283	0.5%	99.9%
More than 100 Apps	89,601	0.1%	100%
Total	**92,329,268**	**100%**	**100%**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] App purchases include paid downloads, subscriptions, and IAP sales.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 22

Summary of Named Consumer Plaintiffs' Installed Apps and Paid/Subscription/IAP App Purchases

August 2016 – October 2021

	[redacted]	[redacted]	[redacted]			[redacted]
Paid/Subscription/IAP Apps Installed	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Paid/Subscription/IAP Installed with Purchases	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Free Apps Installed	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Apps with Ads Installed	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
All Apps Installed	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Number of App Categories, All Installed Apps	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Game App Categories	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Non-Game App Categories	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Total Consumer Spend During Class Period	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Range of Prices Paid	[redacted]	[redacted]				
Device Maker(s)	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Notes:

[1] Only apps that were last updated by the plaintiffs in August 2016 or later are counted as installed.

[2] Consumer spend from July 3, 2021 to October 2021 for [redacted] based on purchase history data rather than transactions data.

[3] A "paid/subscription/IAP app" is an app that has a paid app download and/or offers in-app purchases (in-app monetizations or subscriptions) through the Google Play Store. A "free app" is free to download and does not offer in-app purchases through the Google Play Store.

[4] Information about monetization strategy and app category could not be found for 19 apps that the plaintiffs downloaded after July 2021.

[5] Pre-installed apps are excluded from installed app counts. All cell carrier-, device OEM-, and Google-owned apps are assumed to be pre-installed unless the plaintiffs made purchases from them.

[6] A "purchase" is a transaction of a Google Play App that is charged, delivered, not refunded or canceled, and whose price is greater than zero. Only transactions made in August 2016 or later are included. Test instruments are excluded.

[7] Prices and consumer spend are post-tax.

[8] Transaction amounts made in Euros are converted to USD using the exchange rate from the day on which the transaction occurred.

[9] Devices are identified based on the device makes and models reported in GOOG-PLAY-007203251 for transactions made during the class period and on the makes and models reported in the install data for apps installed during the class period.

[10] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Sources:

[1] Install Data for Named Plaintiffs, GOOG-PLAY-010569458, GOOG-PLAY-007335783, GOOG-PLAY-007335784, GOOG-PLAY-007335785, GOOG-PLAY-007335786, GOOG-PLAY-007335787, GOOG-PLAY-00737956, GOOG-PLAY-00737961, and GOOG-PLAY-007335788

[2] Transactions Data for Named Plaintiffs, GOOG-PLAY-007335789, GOOG-PLAY-007335790, GOOG-PLAY-007335791, GOOG-PLAY-007335792, GOOG-PLAY-007335793, GOOG-PLAY-007335794, GOOG-PLAY-00737958, and GOOG-PLAY-00737959

[3] Purchase History Data for Named Plaintiffs, GOOG-PLAY-010569450

[4] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

[5] Google Transactions Data, GOOG-PLAY-007203251

[6] Google Play Store (accessed October 28, 2021)

[7] Federal Reserve Board of Governors, "Foreign Exchange Rates - H.10", https://www.federalreserve.gov/releases/h10/hist/

Exhibit 23

Distribution of Dollars Spent Across U.S. Consumers

August 2016 - July 3, 2021

Consumer Spend	Number of Consumers	Percent of Consumers	Cumulative Percent of Consumers	Cumulative Percent of Spend
Less than $5	[redacted]	[redacted]	[redacted]	[redacted]
$5 - <$10	[redacted]	[redacted]	[redacted]	[redacted]
$10 - <$25	[redacted]	[redacted]	[redacted]	[redacted]
$25 - <$50	[redacted]	[redacted]	[redacted]	[redacted]
$50 - <$100	[redacted]	[redacted]	[redacted]	[redacted]
$100 - <$200	[redacted]	[redacted]	[redacted]	[redacted]
$200 - <$300	[redacted]	[redacted]	[redacted]	[redacted]
$300 - <$400	[redacted]	[redacted]	[redacted]	[redacted]
$400 - <$500	[redacted]	[redacted]	[redacted]	[redacted]
$500 - <$1,000	[redacted]	[redacted]	[redacted]	[redacted]
$1,000 - <$2,000	[redacted]	[redacted]	[redacted]	[redacted]
$2,000 - <$5,000	[redacted]	[redacted]	[redacted]	[redacted]
$5,000 - <$10,000	[redacted]	[redacted]	[redacted]	[redacted]
$10,000 - <$20,000	[redacted]	[redacted]	[redacted]	[redacted]
$20,000 - <$50,000	[redacted]	[redacted]	[redacted]	[redacted]
$50,000 - <$100,000	[redacted]	[redacted]	[redacted]	[redacted]
$100,000 or More	[redacted]	[redacted]	[redacted]	[redacted]
All Consumers	[redacted]	[redacted]	[redacted]	[redacted]

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Sales revenue (sale_revenue_pre_tax_micros) for each consumer is aggregated across the time period.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 24

Number of Purchases Made by U.S. Consumers

August 2016 – July 3, 2021

Number of Purchases	Number of Consumers	Percent of Consumers	Cumulative Percent of Consumers	Number of Apps Purchased From
1 Purchase	19,628,465	21%	21%	154,861
2 Purchases	9,168,867	10%	31%	142,744
3 Purchases	6,081,287	7%	38%	135,163
4 Purchases	4,576,431	5%	43%	129,918
5 or More Purchases	52,874,218	57%	100%	386,995
All U.S. Consumers	**92,329,268**	**100%**	**100%**	**404,325**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] App purchases include paid downloads, subscriptions, and IAP sales.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 25

[redacted]

Exhibit 26

Number of Putative Developer Class Members by Service Fee Rate Category by Year

2017 – 2021

Among Developers with Service Fee Rate >0%

	2017		2018		2019		2020		2021	
Service Fee Rate	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**
[redacted]										
All Developers	[redacted]									

Notes:

[1] A developer's weighted-average annual service fee rate is calculated as total revenue ("iusd_original_revenue") divided by total consumer spend ("iusd_original_consumer_spend") in the app-level spend data.

[2] Developers with an annual weighted-average service fee rate that is missing, negative, or zero in any year between 2017 and 2021 are excluded.

[3] Service fee rates from net refunds are included in this analysis.

Source:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 27

Number of Putative Developer Class Members by Service Fee Rate Category by Year

2017 – 2021

Among Developers with Google Play Sales and Service Fee Rate >0% in Each Year from 2017-2021

	2017		2018		2019		2020		2021	
Service Fee Rate	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**	**Number**	**Percentage**
[redacted]	[redacted]									
All Developers										

Notes:

[1] A developer's weighted-average annual service fee rate is calculated as total revenue ("iusd_original_revenue") divided by total consumer spend ("iusd_original_consumer_spend") in the app-level spend data.

[2] Developers with an annual weighted-average service fee rate that is missing, negative, or zero in any year between 2017 and 2021 are excluded.

[3] This table only includes developers that have sales in Google Play in each year between 2017 and 2021. There are [redacted] such developers in total out of [redacted] putative class developers.

[4] Service fee rates from net refunds are included in this analysis.

Source:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 28

Published Service Fee Rate Reductions by Certain App Stores

Store	Platform	Year of rate reduction (effective)	Amount of Rate Reduction	Developers Receiving Rate Reduction	Percent of Putative Developer Class Members (if rate reduction were applied to Play)	Percent of Putative Developer Class Members' Spend (if rate reduction were applied to Play)
Amazon Appstore	Android	2018	30% to 20%	Movie and TV subscription app developers	3%	9%
Amazon Appstore	Android	2021 (Q4)	30% to 20% (plus additional rebates as Amazon Web Services Credit)	Developers with less than $1M in sales in the previous calendar year	99.7%	23%
Apple App Store	iOS	2016 (June)	30% to 15%	Subscriptions retained for 12 months or more	<22%	<24%
Apple App Store	iOS	2016	30% to 15%	Premium subscription video developers	3%	9%
Apple App Store	iOS	2021 (Jan)	30% to 15%	Developers with less than $1M in sales in the previous calendar year	99.7%	23%
Microsoft Store	PC and XBOX	2015 (Jan)	Before 2015: 30%: 0-$25,000 sales 20%: $25,000+ sales	(No rate reductions)	8%	99%
			After 2015: 30% flat rate	Developers with at least one app earning over $25,000 in sales received a rate increase	(rate increase)	(rate increase)
Microsoft Store	PC and XBOX	2019 (Mar)	30% to 15%	Non-game, non-Xbox downloads via Microsoft Store search or a Microsoft Store Collection	N/A	N/A
			30% to 5%	Non-game, non-Xbox downloads via developer's direct link and not through Microsoft Store Search or Microsoft Store Collection		
Microsoft Store	PC and XBOX	2020 (Jan)	5% to 15%	Non-game, non-Xbox downloads via developer's direct link and not through Microsoft Store Search or Microsoft Store Collection	N/A	N/A
Microsoft Store	PC and XBOX	2021 (Aug)	30% to 12%	Non-Xbox Games	33%	69%
		2021 (Jul)	15% to 0%	Non-game apps charging through third party payment platform	N/A	N/A
One Store	Android	2018 (Jul)	30% to 20%	Developers who use One Store's payment platform	N/A	N/A
			30% to 5%	Developers who use their own payment platform	N/A	N/A
Steam	PC	2018 (Nov) announced	Before Nov 2018: 30%, with exceptions for small developers in Steam Direct program	All games: Revenue per app is considered, thresholds are total revenues starting from October 1st, 2018		
			After Nov 2018: 30%: $0-$10M		N/A	N/A
			25%: $10M-$50M		0.2%	15%
			20%: $50M+		0.1%	42%

Notes:

[1] For each app store, the percentage of putative developer class members is calculated for the analogous developers if they were to receive the rate reduction in Play over the class period.

[2] The percentages for all Play subscription developers reflect the upper bounds that are shown for rate changes applied to subscribers of 1+ year.

[3] For Amazon Appstore and Apple App Store's 2021 small developers rate reduction, the percentage of putative developer class members who could potentially be affected by the policy is calculated as
(total developers whose annual revenue is <$1M at least once in the class period / total developers over the class period).

[4] For Microsoft Store rate increase in 2015, the percentage of putative developer class members who could potentially be affected by the policy is calculated as (total developers who own at least one app with spend>$25,000 / total developers over the class period).

[5] For Steam rate reduction in 2018, the percentage of putative developer class members who could potentially be affected by the policy is calculated as (total developers who have revenue per game in the applicable threshold / total developers over the class period). The percentage of total consumer spend is calculated as (aggregated game revenue in a rate bracket / total consumer spend over the class period).

[6] Participants in Apple's Video Partner Program in 2016 and Amazon's 2018 rate reduction program are identified as (i) owning an app in the Entertainment category are in Google's LRAP/LRAP++ program, or are Amazon, due to their Prime Video app.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] See Sources exhibit and this report's production for Bates numbers of deal contracts and app package names.

Exhibit 29

Fees for Payment Processing Service Providers

Payment Processor Base Rates (as of November 2021)

Payment Processing Service Provider	Base Rate (U.S.)
PayPal	[redacted]
Braintree (owned by PayPal)	
Stripe	
Square	

Notes:

[1] Effective August 2, 2021, PayPal's base rate is [redacted] for online credit and debit card transactions. For Digital Payments, PayPal's base rate is [redacted] per transaction.

[2] Square's base rate for card-not-present payments is [redacted], and is applied to purchases through Square Online Store, Square Online Checkout, eCommerce API, or online invoice payments. If a card on file is used or the card details are entered manually, the base rate is [redacted].

[3] PayPal and Stripe offer Chargeback Protection services with an additional [redacted] charge per transaction.

Sources:

[1] "PayPal pricing," PayPal, https://www.paypal.com/us/business/pricing, accessed November 17, 2021.

[2] "PayPal Fee Changes: Effective August 2, 2021," PayPal, https://www.paypalobjects.com/marketing/ua/pdf/US/en/feepages-080221.pdf, accessed November 17, 2021.

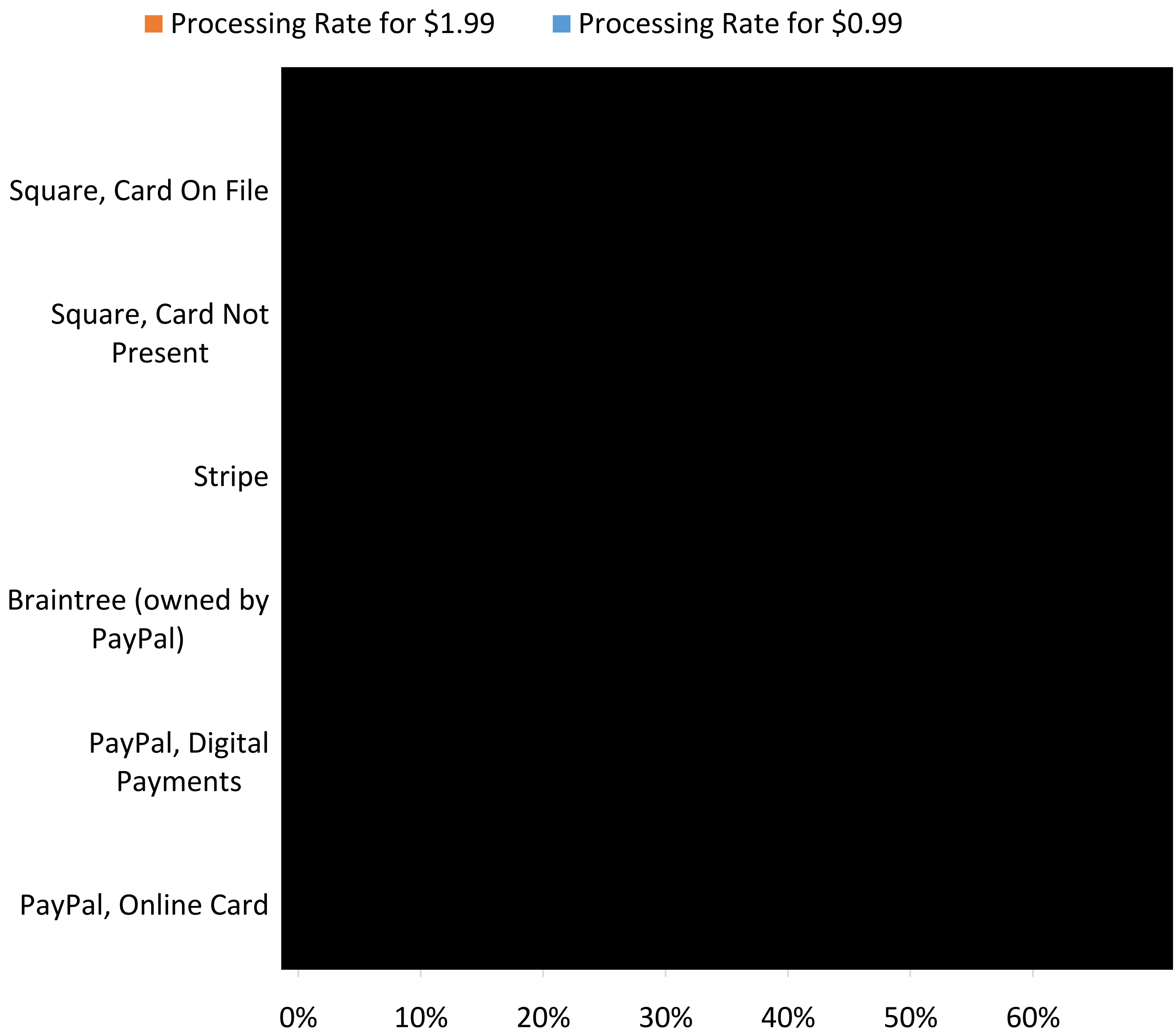
[3] "Pricing," Braintree, https://www.braintreepayments.com/braintree-pricing, accessed November 17, 2021.

[4] "Pricing built for businesses of all sizes," Stripe, https://stripe.com/pricing, accessed November 17, 2021.

[5] "Understanding Square processing fees," Square, https://squareup.com/us/en/payments/our-fees, accessed November 17, 2021.

Exhibit 30

Effective Payment Processing Rates of Alternative Processors at Low Price Points



Source: Exhibit 29.

Exhibit 31

Putative Developer Class Members Selling at Prices of $0.99 and $1.99

August 2016 - July 3, 2021

	Count of Developers	Percent of Developers	Cumulative Percent of Developers
Developers Only Selling at $0.99	10,702	22%	22%
Developers Only Selling at $1.99 or lower (>$0.99)	7,135	14%	36%
Total Putative Developer Class Members	**49,387**	**100%**	**100%**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Developers are identified based on unique Gaia IDs. Gaia IDs are crosswalked from the app-level spend data for putative class developers to the transactions data based on app package name and month and year of purchase. The app-level spend data ranges from August 2016 to December 2021.

[3] Some putative class developer Gaia IDs are not crosswalked with the transactions data because they don't sell to U.S. consumers during the class period.

Sources:

[1] Google Transactions Data, GOOG-PLAY-007203251

[2] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

Exhibit 32

Prices that End in “99” as a Percentage of U.S. Consumer Transactions

August 2016 - July 3, 2021

Retail Price	Percent of Transactions
Price is $0.99	17%
Price is $1.99	14%
Price is $4.99	18%
Price is $9.99	12%
Other Prices Ending in "99"	36%
Prices Not Ending in "99"	3%
Total	**100%**

Note:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 33

Percent Reductions from Price Point to Price Point

	Percent Reduction
$5.99 to $4.99	17%
$4.99 to $3.99	20%
$3.99 to $2.99	25%
$2.99 to $1.99	33%
$1.99 to $.99	50%

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 34

Summary of SKUs with Service Fee Rate Reductions

Rate Change Study	Number of Developers	Number of Apps	Number of SKUs	Share of Consumers Making Purchases	Share of Consumer Spend
Paid App Downloads	4,538	9,264	9,264	18%	1%
Scraped Data	*784*	*1,557*	*1,557*	*14%*	*0.2%*
App-Level Spend Data	*4,499*	*8,991*	*8,991*	*18%*	*0.3%*
Subscriptions	5,229	7,312	17,013	22%	4%
Deal Developers	*68*	*211*	*706*	*8%*	*1%*
Subscription Developers	*5,167*	*7,101*	*16,307*	*15%*	*3%*
IAPs	15,967	37,367	441,383	35%	7%
All SKUs	**22,995**	**51,847**	**467,660**	**53%**	**11%**

Notes:

[1] "Deal Developers" SKUs are associated with ADAP, LRAP, LRAP++, and SwG programs. "Subscription Developers" SKUs are subscriptions and exclude SKUs from Deal Developers. Deal or Subscription Developer SKUs only include SKUs that experienced a rate reduction, identified as the service fee rate fell from 30% to 20% or lower and was sustained for 3+ consecutive months in Aug 2016-July 2021 transactions data. "Scraped Data" and "App-Level Spend Data" SKUs are paid app downloads SKUs that experienced a rate reduction identified in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021. All "Deal Developers" SKUs, "Subscription Developers" SKUs, "Scraped Data" SKUs, and "App-Level Spend Data" SKUs are required to have at least one price data point in both the month before and month after the rate reduction. "IAPs" from Dr. Burtis' report includes all IAPs in the Google Play Store Price Tracking Data (snapshots in June and October 2021 and February 2022) that transacted at least once during the class period in Aug 2016-July 2021 transactions data, experienced a rate reduction for their in-app purchases in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021, and exist in both the June 2021 snapshot and at least one of the October 2021 and February 2022 snapshots.

[2] Number of Developers is based on developer Gaia IDs from the U.S. Developers and U.S. Consumers app-level spend data (GOOG-PLAY-005535885, -86; GOOG-PLAY-010801688, -89).

[3] "Share of Consumers Making Purchases" and "Share of Consumer Spend" shows the share of total consumers and consumer spend of purchases of SKUs with service fee reductions from August 1, 2016 to July 3, 2021 in the transactions data.

[4] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Sources:

[1] Google Play Store Paid App Scrape Data

[2] Google Transactions Data, GOOG-PLAY-007203251

[3] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[4] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

[5] Google Play Store IAP Price Tracking Data, June 2021, October 2021, February 2022, GOOG-PLAY2-000483364

[6] See this report's production for Bates numbers of deal developer contracts

Exhibit 35

Constant-Priced SKUs Among SKUs with Rate Reductions

Rate Change Study	Number of SKUs Considered	Number of SKUs with No Price Change During Class Period	Share of SKUs with No Price Change During Class Period
Paid App Downloads	9,264	1,988	21%
Scraped Data	*1,557*	*1,303*	*84%*
App-Level Spend Data	*8,991*	*1,673*	*19%*
Subscriptions	17,013	14,010	82%
Deal Developers	*706*	*503*	*71%*
Subscription Developers	*16,307*	*13,507*	*83%*
IAPs	441,383	416,892	94%
All SKUs	**467,660**	**432,890**	**93%**

Notes:

[1] “Deal Developers” SKUs are associated with ADAP, LRAP, LRAP++, and SwG programs. “Subscription Developers" SKUs are subscriptions and exclude SKUs from Deal Developers. Deal or Subscription Developer SKUs only include SKUs that experienced a rate reduction, identified as the service fee rate fell from 30% to 20% or lower and was sustained for 3+ consecutive months in Aug 2016-July 2021 transactions data. “Scraped Data” and "App-Level Spend Data" SKUs are paid app downloads SKUs that experienced a rate reduction identified in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021. All “Deal Developers” SKUs, “Subscription Developers" SKUs, "Scraped Data" SKUs, and "App-Level Spend Data" SKUs are required to have at least one price data point in both the month before and month after the rate reduction. "IAPs" from Dr. Burtis' report includes all IAPs in the Google Play Store Price Tracking Data (snapshots in June and October 2021 and February 2022) that transacted at least once during the class period in Aug 2016-July 2021 transactions data, experienced a rate reduction for their in-app purchases in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021, and exist in both the June 2021 snapshot and at least one of the October 2021 and February 2022 snapshots.

[2] For Subscriptions, price changes are evaluated using net prices from the Google transactions data from August 2016 to July 3rd 2021. For paid app downloads in the app-level spend data, prices are evaluated based on prices for unique combinations of form of payment, device, app category, purchase type, app, developer, month, and year in the app-level spend data. For paid app downloads in the Scraped Data, price changes are evaluated based on monthly retail prices from April 2021 to January 2022. For IAPs, price changes are evaluated using list prices reported in snapshots taken in June 2021, October 2021, and February 2022.

[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[4] It is possible for an app to appear in multiple studies and have differing result. This is because their prices and service fees can appear differently in the source data files use by these studies. When calculating the overall share of SKUs in all studies that maintained a constant price, a SKU is considered to have a constant price if and only if its price was found to be constant in all rate change studies it was included in.

Sources:

[1] Google Play Store Paid App Scrape Data
[2] Google Transactions Data, GOOG-PLAY-007203251
[3] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689
[4] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688
[5] Google Play Store IAP Price Tracking Data, June 2021, October 2021, February 2022, GOOG-PLAY2-000483364
[6] See this report’s production for Bates numbers of deal developer contracts

Exhibit 36

Summary of Pass-Through Rates Found by Dr. Burtis

1 Month Before vs. 1 Month After
August 2016 – February 2022

	SKUs	
Purchase Type	**Total**	**Share w/Positive Pass-Through**
Paid App Downloads		
Scraped Data	1,557	1%
App-Level Spend Data	8,991	13%
Subscriptions		
Deal Developers	706	2%
Subscription Developers	16,307	1%
IAPs	441,383	2%

Notes:

[1] “Deal Developers” SKUs are associated with ADAP, LRAP, LRAP++, and SwG programs. “Subscription Developers" SKUs are subscriptions and exclude SKUs from Deal Developers. Deal or Subscription Developer SKUs only include SKUs that experienced a rate reduction, identified as the service fee rate fell from 30% to 20% or lower and was sustained for 3+ consecutive months in Aug 2016-July 2021 transactions data. “Scraped Data” and "App-Level Spend Data" SKUs are paid app downloads SKUs that experienced a rate reduction identified in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021. All “Deal Developers” SKUs, “Subscription Developers" SKUs, "Scraped Data" SKUs, and "App-Level Spend Data" SKUs are required to have at least one price data point in both the month before and month after the rate reduction. "IAPs" from Dr. Burtis' report includes all IAPs in the Google Play Store Price Tracking Data (snapshots in June and October 2021 and February 2022) that transacted at least once during the class period in Aug 2016-July 2021 transactions data, experienced a rate reduction for their in-app purchases in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021, and exist in both the June 2021 snapshot and at least one of the October 2021 and February 2022 snapshots.

[2] Price changes for IAPs are identified by comparing prices in June 2021 to average prices in October 2021 and February 2022. Price changes for "Scraped Data" SKUs are identified by comparing the retail prices in the scraped data in the month before and the month after the rate change occurred. Price changes for all other purchase types are identified by comparing weighted average net prices in the month before the rate change occurred to the month after.

[3] A SKU has "positive pass-through" if its price decreased after the rate change.

Sources:

[1] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688
[2] Google Play Store Paid App Scrape Data
[3] Google Transactions Data, GOOG-PLAY-007203251
[4] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689
[5] Google Play Store IAP Price Tracking Data, June 2021, October 2021, February 2022, GOOG-PLAY2-000483364
[6] See this report’s production for Bates numbers of deal developer contracts

Exhibit 37

Counts of U.S. Consumers in the Putative Class, By Maximum Price Point of Transactions

August 2016 - July 3, 2021

	Count of Consumers	Percent of Consumers	Cumulative Percent of Consumers
Consumers Only Buying at a Max of $0.99	4,868,989	5%	5%
Consumers Only Buying at a Max of $1.99	3,667,004	4%	9%
Consumers Only Buying at a Max of $2.99	4,062,814	4%	14%
Consumers Buying at above $2.99	79,730,461	86%	100%
Total Consumers	**92,329,268**	**100%**	**100%**

Note:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 38

Count of U.S. Consumers By Form of Payment

August 2016 – July 3, 2021

Form of Payment	Count of U.S. Consumers	Percent of U.S. Consumers
Card	69,612,363	75%
Gift Cards	22,909,292	25%
PayPal	13,334,243	14%
Direct Carrier Billing	10,210,611	11%
All U.S. Consumers	**92,329,268**	**100%**

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Counts include U.S. consumers that ever used a payment method between August 1, 2016 and July 3, 2021. Consumers may have used multiple different forms of payment during the class period.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 39

Named Consumer Plaintiffs' Spend on Security Apps

August 2016 – October 2021

Named Consumer Plaintiff	Security App Name	Consumer Spend
[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]
All Named Consumer Plaintiffs	**Total Spend on Security Apps**	[redacted]

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded. Only transactions made in August 2016 or later are included.

[2] Consumer spend from July 3, 2021 to October 2021 for [redacted] is based on purchase history data rather than transactions data.

[3] A security app is an app that protects consumers' transactions online and/or prevents consumers from receiving malicious content.

[4] [redacted]

[5] Prior to the class period, plaintiff [redacted]

[6] Security apps are manually identified based on app name and information available on the Google Play Store.

[7] Purchase dates are based on transaction data for named plaintiffs.

[8] Consumer spend includes spend on paid app downloads and in-app purchases (subscriptions and in-app monetizations). Consumer spend also includes tax.

Sources:

[1] Google Transactions Data, GOOG-PLAY-007203251.

[2] Transactions Data for Named Plaintiffs, GOOG-PLAY-007335789, GOOG-PLAY-007335790, GOOG-PLAY-007335791, GOOG-PLAY-007335792, GOOG-PLAY-007335793, GOOG-PLAY-007335794, GOOG-PLAY-00737958, and GOOG-PLAY-00737959

[3] Purchase History Data for Named Plaintiffs, GOOG-PLAY-010569450

[4] https://play.google.com/store/apps/details?id=com.kii.safe&hl=en_US&gl=US

[5] https://play.google.com/store/apps/details?id=com.att.callprotect&hl=en_US&gl=US

[6] https://play.google.com/store/apps/details?id=com.symantec.mobilesecurity&hl=en_US&gl=US

[7] https://play.google.com/store/apps/details?id=ch.threema.app&hl=en_US&gl=US

Exhibit 40

Top 10 Security Apps by U.S. Consumer Spend

August 2016 - July 3, 2021

Rank	App Name	Description of Features	App Category	Total Consumer Spend	Number of U.S. Consumers	Total Installs	Price	Has Ads?	IAP Price Range
1	NordVPN	NordVPN provides private and secure internet access to users worldwide through a VPN.	Tools	[redacted]	[redacted]	10,000,000+	Free	No	$7.99/month - $84.99/year
2	RoboKiller	Robokiller is a spam call and robocall blocker that blocks and blacklists incoming spam calls.	Communication	[redacted]	[redacted]	5,000,000+	Free	No	$0.99/month - $39.99/year
3	Lookout	Lookout provides premium mobile security and identity protection for Android devices.	Tools	[redacted]	[redacted]	100,000,000+	Free	No	$1.99/month - $99.99/year
4	Keepsafe Photo Vault	Keepsafe allows users to protect personal photos and videos in a PIN-protected and encrypted vault.	Photography	[redacted]	[redacted]	50,000,000+	Free	Yes	$0.99/month - $299.99/lifetime
5	Norton Secure VPN	Norton Secure VPN provides private and secure internet access to users worldwide through a VPN.	Tools	[redacted]	[redacted]	10,000,000+	Free	No	$29.99/year - $39.99/year
6	Keeper Password Manager	Keeper lets users store and manage passwords within an encrypted vault that they can access across devices.	Productivity	[redacted]	[redacted]	10,000,000+	Free	No	$9.99/year - $199.99/lifetime
7	AVG AntiVirus	AVG AntiVirus protects users from receiving viruses and malware. It scans incoming files, apps, and websites for malware, assesses WiFi networks for security, and alerts users if their data has been compromised.	Tools	[redacted]	[redacted]	100,000,000+	Free	Yes	$0.99/month - $89.99/year
8	Norton™ 360	Norton 360 protects users from security threats by scanning incoming files, websites, and apps and assessing WiFi networks for security. Norton 360 also includes the Norton Secure VPN.	Tools	[redacted]	[redacted]	50,000,000+	Free	No	$0.99/month - $104.99/year
9	Hotspot Shield Free VPN	Hotspot Shield Free VPN provides private and secure internet access to users worldwide through a VPN.	Tools	[redacted]	[redacted]	100,000,000+	Free	Yes	$0.97/month - $99.99/lifetime
10	Avast Antivirus	Avast Antivirus protects users from receiving viruses and other types of malware. It also includes an ecrypted Vault for photos.	Tools	[redacted]	[redacted]	100,000,000+	Free	Yes	$0.99/month - $79.99/year

Notes:

[1] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[2] Top security apps are identified based on information available on the Play App Catalog and information available on the Google Play Store. A "security app" is an app that safeguards a user's data and/or blocks the user from receiving malware.

[3] Security apps are ranked by total consumer spend between August 2016 and July 2021.

[4] Total installs, app category, price, and "Has Ads?" are based on the Google Play Store. Developers self report whether they offer ads.

[5] "Price" refers to the price of the paid app download.

[6] "IAPs" include subscriptions and in-app monetizations. IAP price ranges are based on list prices of transactions made between August 2016 and July 2021. Billing periods are determined based on the subscription period recurring unit. If an app offered different SKUs at the same list price but for with different billing period durations, the billing period duration at which the greatest amount of consumer spend was transacted is shown in the price range.

Sources:

[1] Google Transactions Data, GOOG-PLAY-007203251

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

[3] https://play.google.com/store/apps/details?id=com.lookout&hl=en_US&gl=US

[4] https://play.google.com/store/apps/details?id=com.robokiller.app&hl=en_US&gl=US

[5] https://play.google.com/store/apps/details?id=com.nordvpn.android&hl=en_US&gl=US

[6] https://play.google.com/store/apps/details?id=com.kii.safe&hl=en_US&gl=US

[7] https://play.google.com/store/apps/details?id=com.symantec.securewifi&hl=en_US&gl=US

[8] https://play.google.com/store/apps/details?id=com.callpod.android_apps.keeper&hl=en_US&gl=US

[9] https://play.google.com/store/apps/details?id=com.antivirus&hl=en_US&gl=US

[10] https://play.google.com/store/apps/details?id=hotspotshield.android.vpn&hl=en_US&gl=US

[11] https://play.google.com/store/apps/details?id=com.symantec.mobilesecurity&hl=en_US&gl=US

[12] https://play.google.com/store/apps/details?id=com.avast.android.mobilesecurity&hl=en_US&gl=US

Exhibit 41

Putative Developer Class Members with Free Apps

August 2016 - December 2021

	Count of Developers	% of Developers
Putative Developer Class Members with no Free Apps	34,463	70%
Putative Developer Class Members with any Free Apps	14,868	30%
Total	49,331	100%

Notes:

[1] There are apps in the app-level dataset that have more than one GAIA ID associated with it. Because of this, some developers are not accounted for. Some putative class developers have an unknown catalog id.

[2] When there is more than one GAIA ID associated with an app, or more than one GAIA ID to catalog ID combination, then the most recent id, or combination, is kept.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

[3] Global Google Admob Data, GOOG-PLAY-009908838

[4] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 42

Putative Developer Class Members' Apps With Lifespan of 12 Months or Less

Among Apps with Spend in August 2016 - December 2021

First Publish Year	Putative Developer Class Members' Apps	Putative Developer Class Members' Apps with Lifespan of 12 Months or Less	Percentage
2016	14,558	4,069	28%
2017	14,899	6,275	42%
2018	11,079	4,850	44%
2019	10,220	4,607	45%
2020	9,891	5,315	54%
2016 - 2020	**60,647**	**25,116**	**41%**

Notes:

[1] Lifespan of an app is defined as (last month in the transactions data - min(first month in the transactions data, first publish date)).

[2] Analysis is restricted to apps that were transacted during the class period and to developers with publish years between 2016 and 2020. The lifespan is not limited to the class period.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 43

Number of Putative Developer Class Members Remaining with App Sales

Based on Elapsed Time between Epic Store's Launch and Microsoft Store's Rate Reduction to 12%

	Count	Percent
Developers with No Sales on or after April 2019	18,699	38%
Developers with Sales on or after April 2019	30,688	62%
Total Putative Class Developers	**49,387**	**100%**

Note:

[1] The Epic Games Store was launched in December 2018, and the Microsoft Store reduced its rates 32 months later in August 2021. April 2019 is chosen here because it is 32 months after the beginning of the class period.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] https://techcrunch.com/2018/12/06/epic-games-store/ (accessed March 24, 2022).

[3] https://www.protocol.com/bulletins/microsoft-store-commission-cut (accessed March 24, 2022).

Exhibit 44

Sample Calculation of Singer's Pass-Through Rates for Developers Selling Subscriptions

January 2019

App Developer GAIA ID	App Developer Name	App Package Name	App Category	Month	Total Subscription Units Sold in App Category	Developer Subscription Units Sold	Developer's Share of Total Subscription Units Sold	Pass-Through Rate for Subscriptions (1 - Share)	List Price
[redacted]	[redacted]	[redacted]	Lifestyle	Jan. 2019	669,842	59	0.01%	99.99%	$4.99
[redacted]	[redacted]	[redacted]	Game	Jan. 2019	935,919	2	0.0002%	99.9998%	$9.99
[redacted]	[redacted]	[redacted]	Books and Reference	Jan. 2019	56,488	144	0.3%	99.7%	$8.99

Notes:

[1] In the class period, developers included in this table only sold subscriptions through one app in a particular app category. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.

[2] [redacted] sold two subscription SKUs, both priced at $8.99 throughout the class period.

Sources:

[1] Singer backup for Tables 7 & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886

[3] Google Transactions Data, GOOG-PLAY-007203251

[4] Williams backup for Figure 3

Exhibit 45

Dr. Singer's Pass-Through Rates for Example Developers Selling Subscriptions

August 2016 – December 2020

Month	Pass-Through-Rate for Subscriptions		
	[redacted]		
Aug. 2016	99.998%	99.99%	82.6%
Sep. 2016	99.999%	99.99%	86.7%
Oct. 2016	99.996%	99.98%	90.1%
Nov. 2016	99.999%	99.996%	92.0%
Dec. 2016	99.998%	99.997%	96.7%
Jan. 2017	99.89%	99.997%	97.7%
Feb. 2017	99.96%	99.996%	99.0%
Mar. 2017	99.97%	99.99%	99.1%
Apr. 2017	99.99%	99.99%	99.3%
May. 2017	99.997%	99.99%	99.8%
Jun. 2017	99.998%	99.997%	99.8%
Jul. 2017	99.999%	99.997%	99.9%
Aug. 2017	99.999%	99.999%	99.9%
Sep. 2017	99.999%	99.998%	99.8%
Oct. 2017	99.999%	99.997%	99.7%
Nov. 2017	99.9998%	99.999%	99.7%
Dec. 2017		99.999%	99.7%
Jan. 2018	99.98%	99.999%	99.7%
Feb. 2018	99.99%	99.999%	99.6%
Mar. 2018	99.99%	99.999%	99.7%
Apr. 2018	99.998%	99.998%	99.6%
May. 2018	99.9996%	99.998%	99.7%
Jun. 2018	99.999%	99.999%	99.7%
Jul. 2018	99.999%	99.999%	99.7%
Aug. 2018		99.9998%	99.7%
Sep. 2018	99.9996%	99.9998%	99.6%
Oct. 2018	99.999%	99.999%	99.6%
Nov. 2018		99.9997%	99.7%
Dec. 2018		99.9999%	99.7%
Jan. 2019	99.99%	99.9998%	99.7%
Feb. 2019	99.996%		99.8%
Mar. 2019	99.997%	99.9999%	99.9%
Apr. 2019	99.999%	99.999%	99.8%
May. 2019		99.999%	99.8%
Jun. 2019	99.999%	99.9995%	99.9%
Jul. 2019	99.9997%	99.9996%	99.9%
Aug. 2019		99.9999%	99.8%
Sep. 2019	99.9998%		99.8%
Oct. 2019	99.9998%	99.9998%	99.8%
Nov. 2019			99.8%
Dec. 2019			99.8%
Jan. 2020	99.99%		99.9%
Feb. 2020	99.998%		99.9%
Mar. 2020	99.999%		99.9%
Apr. 2020	99.9995%		99.9%
May. 2020			99.9%
Jun. 2020	99.9995%	99.9999%	99.9%
Jul. 2020		99.9998%	99.9%
Aug. 2020		99.9999%	99.9%
Sep. 2020	99.9998%		99.9%
Oct. 2020	99.9998%	99.9999%	99.9%
Nov. 2020			99.9%
Dec. 2020			99.9%
Minimum	**99.9%**	**99.98%**	**82.6%**
Maximum	**99.9998%**	**99.9999%**	**99.9%**

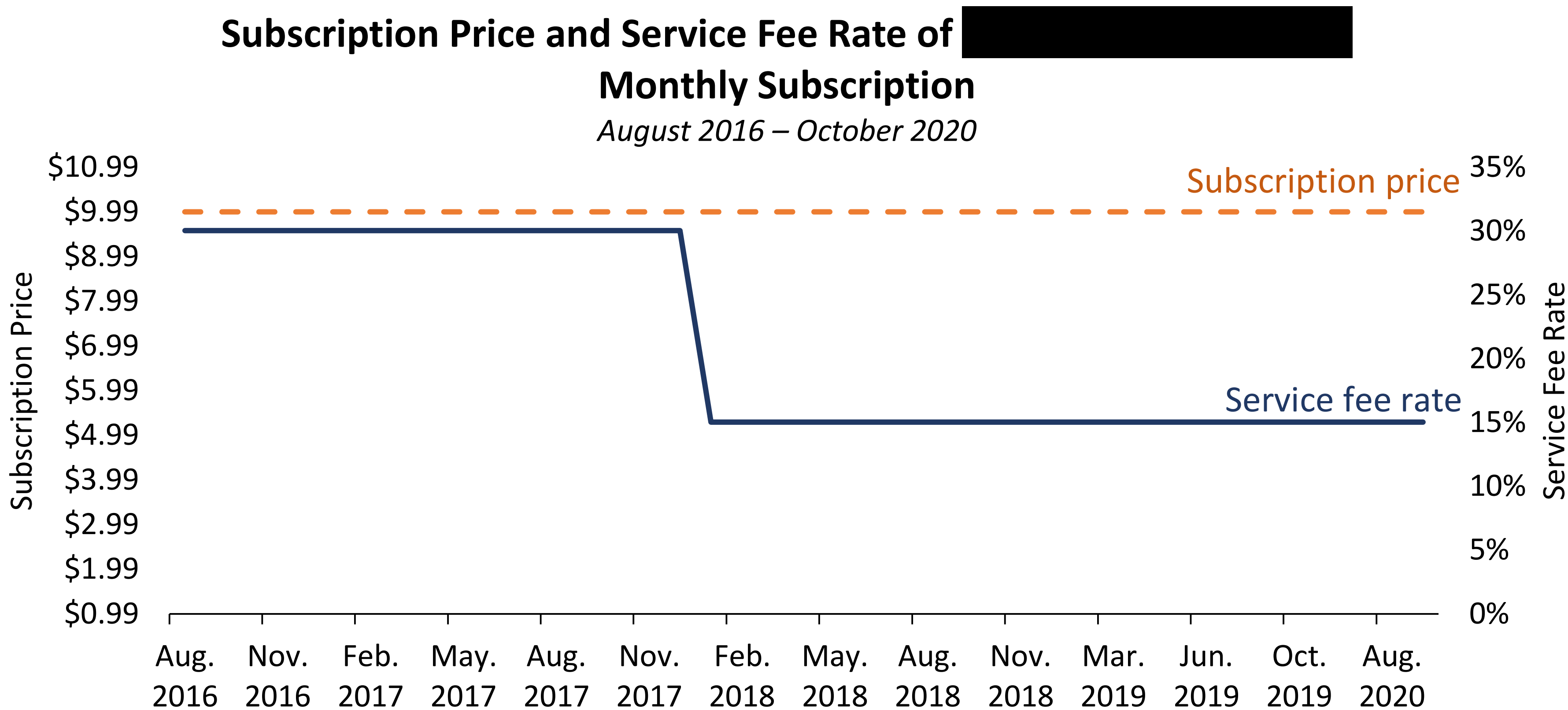
Notes:

[1] In the class period, developers included in this table only sold subscriptions through one app in a particular app category. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.

Sources:

[1] Singer backup for Tables 5, 7, & 8
[2] App-level spend data, Consumers, GOOG-PLAY-005535886
[3] Williams backup for Figure 3

Exhibit 46



Notes:
[1] This graph shows monthly prices and service fee rates for SKU [redacted].
[2] In the class period, [redacted] only sold subscriptions through one app in the Game category. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.
[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.
Sources:
[1] Google Transactions Data, GOOG-PLAY-007203251
[2] Williams backup for Figure 3

Exhibit 47

Summary of Dr. Singer's Pass-Through Rates

January 2012 – December 2020

	Number of Pass-Through Rates	Share of Pass-Through Rates
Pass-Through Rate >99.5%	4,110,126	95%
Pass-Through Rate >0%	4,323,410	99.993%
Pass-Through Rate = 0%	282	0.007%
All Pass-Through Rates	**4,323,692**	**100%**

Note:

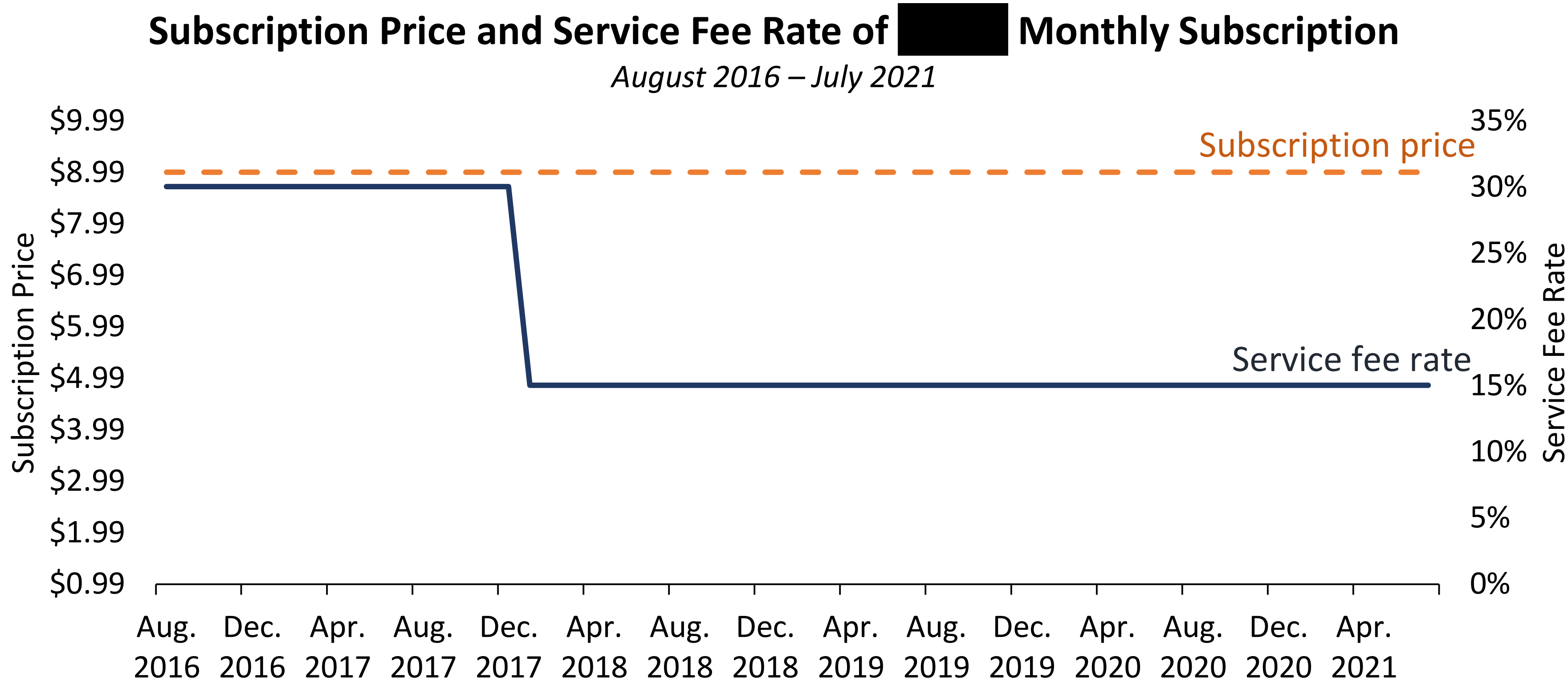
[1] Dr. Singer's pass-through rates are calculated on a unique developer, app category, monetization type, and month level.

Sources:

[1] Singer backup for Tables 7 & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886

Exhibit 48

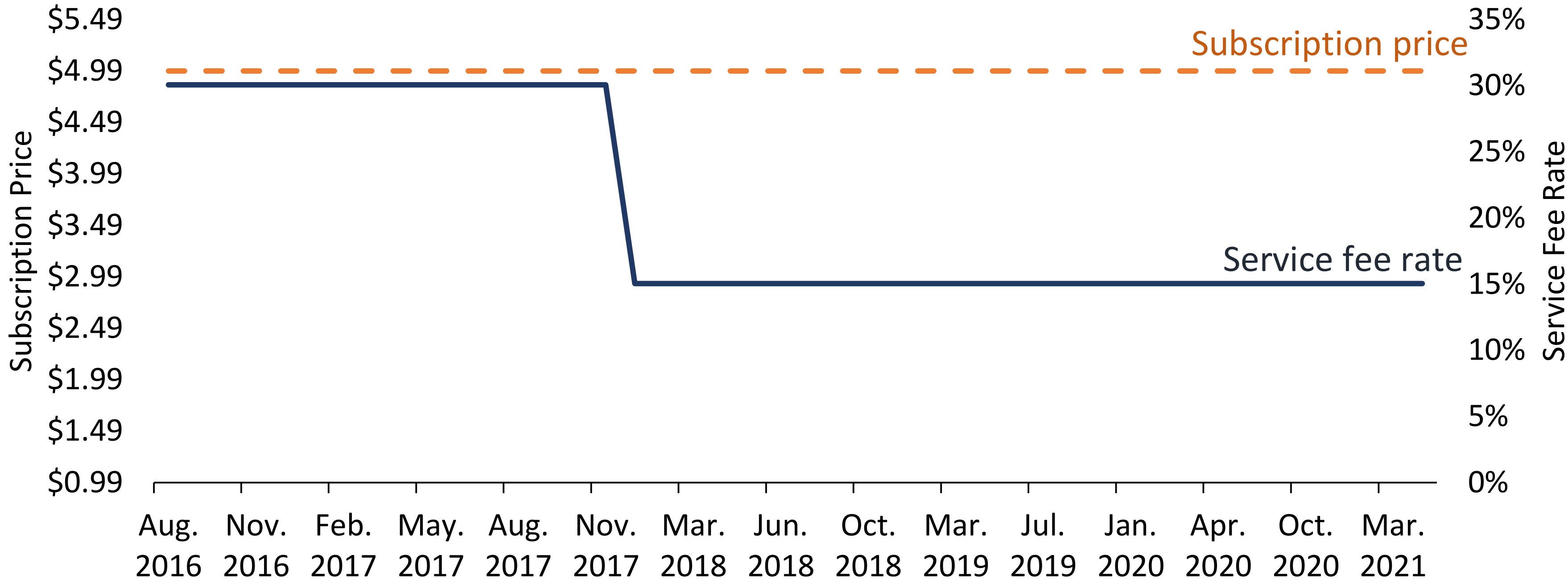


Notes:
[1] [redacted] offers two distinct monthly subscription SKUs, both priced at $8.99 throughout the class period. This graph shows monthly prices and service fee rates for SKU [redacted]. SKU [redacted] had identical prices and service fee rates during the class period, though it was not transacted in July 2021.
[2] In the class period, [redacted]. only sold subscriptions through one app in the Books and Reference category. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.
[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.
Sources:
[1] Google Transactions Data, GOOG-PLAY-007203251
[2] Williams backup for Figure 3

Exhibit 49

Subscription Price and Service Fee Rate of [redacted] Monthly Subscription

August 2016 – April 2021



Notes:
[1] This graph shows monthly prices and service fee rates for SKU [redacted]
[2] In the class period, [redacted] only sold subscriptions through one app in the Lifestyle category. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.
[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.
Sources:
[1] Google Transactions Data, GOOG-PLAY-007203251
[2] Williams backup for Figure 3

Exhibit 50

Summary of Pass-Through Rates Found by Dr. Burtis

6 Months Before vs. 6 Months After
August 2016 – February 2022

Purchase Type	SKUs Total	SKUs Share w/Positive Pass-Through
Paid App Downloads		
Scraped Data	1,557	5%
App-Level Spend Data	8,991	22%
Subscriptions		
Deal Developers	706	6%
Subscription Developers	16,307	2%
IAPs	441,383	2%

Note:

[1] "Deal Developers" SKUs are associated with ADAP, LRAP, LRAP++, and SwG programs. "Subscription Developers" SKUs are subscriptions and exclude SKUs from Deal Developers. Deal or Subscription Developer SKUs only include SKUs that experienced a rate reduction, identified as the service fee rate fell from 30% to 20% or lower and was sustained for 3+ consecutive months in Aug 2016-July 2021 transactions data. "Scraped Data" and "App-Level Spend Data" SKUs are paid app downloads SKUs that experienced a rate reduction identified in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021. All "Deal Developers" SKUs, "Subscription Developers" SKUs, "Scraped Data" SKUs, and "App-Level Spend Data" SKUs are required to have at least one price data point in both the month before and month after the rate reduction. "IAPs" from Dr. Burtis' report includes all IAPs in the Google Play Store Price Tracking Data (snapshots in June and October 2021 and February 2022) that transacted at least once during the class period in Aug 2016-July 2021 transactions data, experienced a rate reduction for their in-app purchases in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021, and exist in both the June 2021 snapshot and at least one of the October 2021 and February 2022 snapshots.

[2] Price changes for IAPs are identified by comparing prices in June 2021 to average prices in October 2021 and February 2022. Price changes for "Scraped Data" SKUs are identified by comparing the simple average of the retail prices in the scraped data in the six months before and six months after the rate change occurred, subject to data availability. Price changes for all other purchase types are identified by comparing weighted average net prices six months before and six months after the rate change occurred, subject to data availability.

[3] A SKU has "positive pass-through" if its price decreased after the rate change.

Sources:

[1] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

[2] Google Play Store Paid App Scrape Data

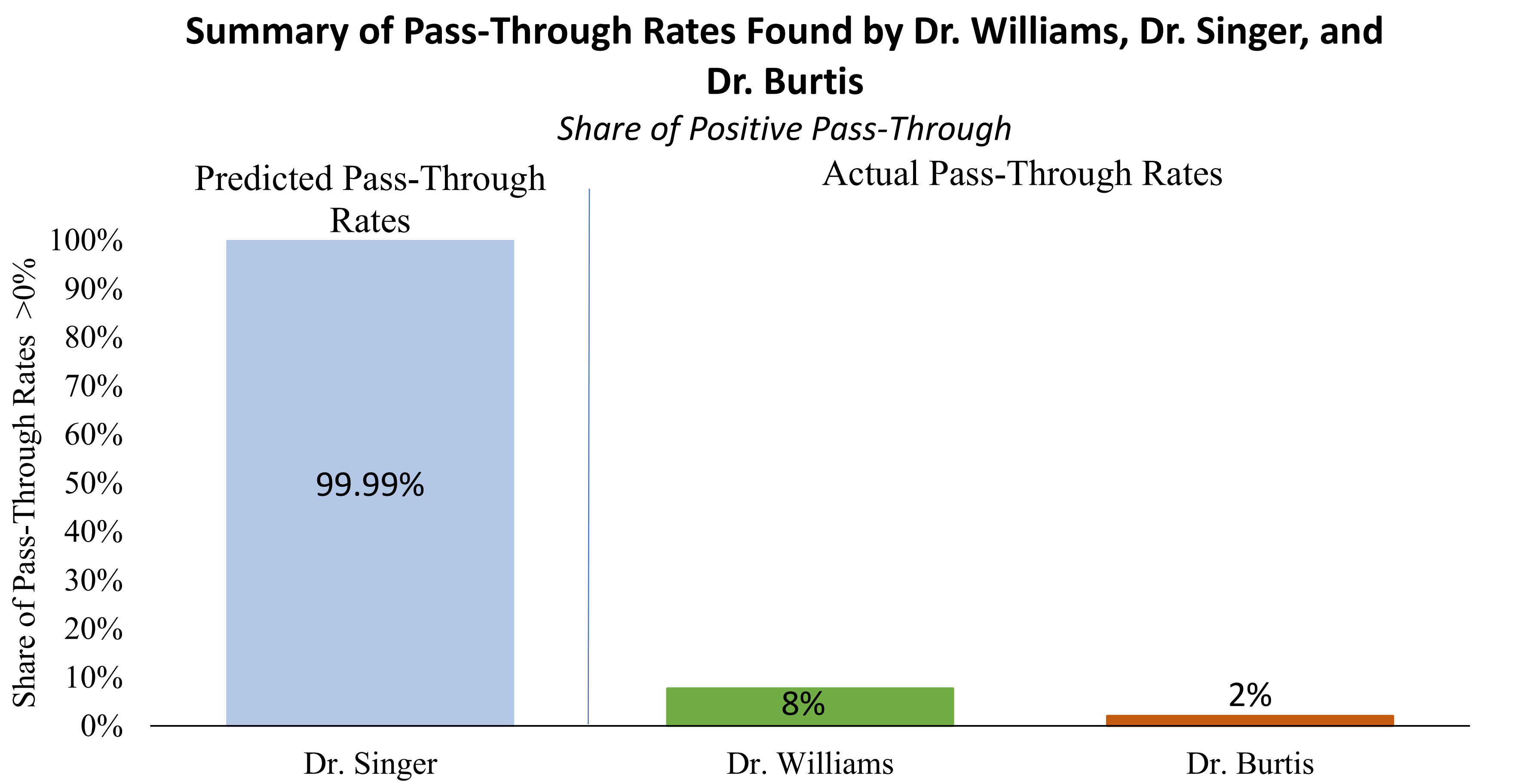
[3] Google Transactions Data, GOOG-PLAY-007203251

[4] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[5] Google Play Store IAP Price Tracking Data, June 2021, October 2021, February 2022, GOOG-PLAY2-000483364

[6] See this report's production for Bates numbers of deal developer contracts

Exhibit 51



Notes:
[1] See notes for Exhibit 52.

Sources:
[2] See sources for Exhibit 52.

Exhibit 52

Summary of Pass-Through Rates Found by Dr. Singer, Dr. Williams, and Dr. Burtis

August 2016 – February 2022

Expert Report	Purchase Type	Developer-Purchase Type-App Category-Months: Total	Developer-Purchase Type-App Category-Months: Share w/Positive Pass-Through	SKUs: Total	SKUs: Share w/Positive Pass-Through
Dr. Singer	Paid App Downloads	180,867	99.997%	n/a	n/a
	Subscriptions	269,800	99.99%	n/a	n/a
	IAPs	449,323	100%	n/a	n/a
Dr. Williams	Paid App Downloads	n/a	n/a	2,562	23%
	Subscriptions	n/a	n/a	34,081	4%
	IAPs	n/a	n/a	39,266	10%
Dr. Burtis	Paid App Downloads				
	Scraped Data	n/a	n/a	1,557	1%
	App-Level Spend Data	n/a	n/a	8,991	13%
	Subscriptions				
	Deal Developers	n/a	n/a	706	2%
	Subscription Developers	n/a	n/a	16,307	1%
	IAPs	n/a	n/a	441,383	2%

Notes:

[1] Dr. Singer's pass-through rates are calculated at a unique developer, app category, purchase type, and month level. Dr Burtis' and Dr. Williams' pass-through rates are calculated at a SKU level.

[2] “Deal Developers” SKUs are associated with ADAP, LRAP, LRAP++, and SwG programs. “Subscription Developers" SKUs are subscriptions and exclude SKUs from Deal Developers. Deal or Subscription Developer SKUs only include SKUs that experienced a rate reduction, identified as the service fee rate fell from 30% to 20% or lower and was sustained for 3+ consecutive months in Aug 2016-July 2021 transactions data. “Scraped Data” and "App-Level Spend Data" SKUs are paid app downloads SKUs that experienced a rate reduction identified in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021. All “Deal Developers” SKUs, “Subscription Developers" SKUs, "Scraped Data" SKUs, and "App-Level Spend Data" SKUs are required to have at least one price data point in both the month before and month after the rate reduction. "IAPs" from Dr. Burtis' report includes all IAPs in the Google Play Store Price Tracking Data (snapshots in June and October 2021 and February 2022) that transacted at least once during the class period in Aug 2016-July 2021 transactions data, experienced a rate reduction for their in-app purchases in the app-level spend data from a rate of 30% to 20% or lower in or after July 2021, and exist in both the June 2021 snapshot and at least one of the October 2021 and February 2022 snapshots.

[3] Only apps included in Dr. Burtis' and Dr. Williams' analyses are included in counts of Dr. Singer's pass-through rates for developer-purchase type-app category-month combinations. Only pass-through rates calculated by Dr. Singer for months in the class period and for apps that are not Google- or developer-owned are included.

[4] "Positive pass-through" in Dr. Burtis' rate change studies means that the SKU price decreased after a rate change. Price changes for IAPs are identified by comparing prices in June 2021 to average prices in October 2021 and February 2022. Price changes for "Scraped Data" SKUs are identified by comparing retail prices in the scraped data in the month before and the month after the rate change occurred. Price changes for all other purchase types are identified by comparing weighted average net prices in the month before the rate change occurred to the month after.

Sources:

[1] Singer backup for Tables 7 & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

[3] Williams backup for Tables 5, 6, & 7

[4] Google Transactions Data, GOOG-PLAY-007203251

[5] Google Play Store Paid App Scrape Data

[6] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[7] Google Play Store IAP Price Tracking Data, June 2021, October 2021, February 2022, GOOG-PLAY2-000483364

[8] See this report’s production for Bates numbers of deal developer contracts

Exhibit 53

Comparison of Dr. Singer's "Implied Pass-Through Rates" and His Predicted Pass-Through Rates

App	Product	Website Price	Google Play Store Price	Dr. Singer's "Implied" Pass-Through Rate	Dr. Singer's Predicted Pass-Through Rate
[redacted]		$29.99/year	$29.99/year	0%	99%
		$4.99/month	$4.99/month	0%	90%
		$7.99/month	$7.99/month	0%	62%
		$1.99/month	$1.99/month	0%	71%
		$7.49	$7.49	0%	85%
		$4.99/month	$4.99/month	0%	23%
		$4.99/month	$4.99/month	0%	97%

Note:

[1] Dr. Singer's predicted pass-through rate is the volume weighted average pass-through rate (averaged across month) at the app package name - monetization type - app category level, based on his backup data for Tables 7 and 8 in his report, restricted to August 2016 to December 2020. Dr. Singer's predicted pass-through rate is calculated at the app developer - monetization type - app category level for each year-month.

Sources:

[1] Price for [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[2] Price for [redacted] on the developer's website [redacted] accessed with an Android deviceas of March 24, 2022.

[3] Price for [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[4] Price for [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[5] Price of the Android mobile version of [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[6] Price for [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[7] Price for [redacted] on the developer's website [redacted] accessed with an Android device as of March 24, 2022.

[8] Singer backup for Tables 7 and 8.

Exhibit 54

Dr. Singer's Pass-Through Rates vs. Pass-Through Rates Based on All App Categories for Select Game Apps

January 2012 – December 2020

App Name	App Package Name	App Category	Dr. Singer's Original Pass-Through Rate	Pass-Through Rate, All App Categories
[redacted]				

Note:

[1] "Dr. Singer's Original Pass-Through Rate" is the volume-weighted average of Dr. Singer's original pass-through rate across all monetization types and months. Singer combines all gaming apps under the app category "Game" prior to calculating pass-through rates on a developer-category-monetization type-month level. "Pass-Through Rate, All App Categories" is the volume-weighted average of the pass-through rate calculated based using granular game categories across all monetization types and months.

Sources:

[1] Singer backup for Tables 7 & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886 and GOOG-PLAY-010801688

Exhibit 55

Dr. Singer's Pass-Through Rates, All App Categories

January 2012 – December 2020

App Category	Pass-Through Rate
ALL	**76.6%**
ART_AND_DESIGN	67.3%
AUTO_AND_VEHICLES	82.4%
BEAUTY	25.0%
BOOKS_AND_REFERENCE	80.4%
BUSINESS	81.6%
COMICS	55.6%
COMMUNICATION	86.6%
DATING	82.3%
EDUCATION	92.1%
ENTERTAINMENT	83.6%
EVENTS	62.0%
FINANCE	69.9%
FOOD_AND_DRINK	77.3%
GAME_ACTION	86.5%
GAME_ADVENTURE	68.5%
GAME_ARCADE	86.8%
GAME_BOARD	73.6%
GAME_CARD	88.0%
GAME_CASINO	92.3%
GAME_CASUAL	51.6%
GAME_EDUCATIONAL	71.1%
GAME_MUSIC	71.4%
GAME_PUZZLE	88.1%
GAME_RACING	82.9%
GAME_ROLE_PLAYING	94.9%
GAME_SIMULATION	92.8%
GAME_SPORTS	76.5%
GAME_STRATEGY	83.3%
GAME_TRIVIA	57.7%
GAME_WORD	80.4%
HEALTH_AND_FITNESS	94.7%
HOUSE_AND_HOME	59.7%
LIBRARIES_AND_DEMO	57.9%
LIFESTYLE	67.9%
MAPS_AND_NAVIGATION	82.0%
MEDIA_AND_VIDEO	88.4%
MEDICAL	92.6%
MUSIC_AND_AUDIO	40.7%
NEWS_AND_MAGAZINES	85.9%
PARENTING	79.1%
PERSONALIZATION	90.3%
PHOTOGRAPHY	90.7%
PRODUCTIVITY	88.6%
SHOPPING	47.4%
SOCIAL	89.2%
SPORTS	80.6%
TOOLS	95.8%
TRANSPORTATION	91.8%
TRAVEL_AND_LOCAL	89.0%
VIDEO_PLAYERS	87.7%
WEATHER	79.5%

Note:

[1] The “Family” game category is excluded because it is not currently available on Google Play Store. In the app-level spend data (GOOG-PLAY-005535886), there are only 3 apps belonging to this category without any records after 2018.

Sources:

[1] App-level spend data, Consumers, GOOG-PLAY-005535886

[2] Singer backup for Tables 7 & 8

Exhibit 56

Share of Developers Abandoning "99" Pricing in But-For World

August 2016 – December 2020

Count of Developers Using "99" Pricing in Actual World	80,103
Count of Developers Abandoning "99" Pricing in But-For World	79,984
Share of Developers Abandoning "99" Pricing in But-For World	99.9%

Notes:

[1] This table includes all developers that 1) sell to U.S. consumers, 2) Singer calculated a pass-through rate for and the pass-through rate could be crosswalked from Singer's data to the transactions data on app package name, monetization type (in-app purchase, subscription, or paid app download), purchase month and year, and app category, and 3) have only sold at list prices ending with ".99".

[2] "Count of Developers Abandoning "99" Pricing in But-For World" includes all developers that would price at least one SKU at a list price that does not end in ".99" if Singer's but-for service fee rates were in place. But-for list prices are calculated as the actual list price minus the but-for dollar reduction in the service fee rate multiplied by the pass-through rate for the respective month, app, app category, and monetization type.

[3] Government- and Google-owned apps are excluded.

[4] Developers are counted by Gaia ID.

Sources:

[1] Singer backup for Tables 3, 5, 7, & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886

[3] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 57

Sample Calculation of Singer's Pass-Through Rates for Developers Selling Subscriptions

June 2017

App Developer GAIA ID	App Developer Name	App Package Name	App Category	Month	Total Subscription Units Sold in App Category	Developer Subscription Units Sold	Developer's Share of Total Subscription Units Sold	Pass-Through Rate for Subscriptions (1 - Share)	List Price	But-for Price According to Singer's Pass-through & But-for Rate
[redacted]			Lifestyle	Jun. 2017	[redacted]				$4.99	[redacted]

Notes:

[1] In the class period, [redacted]. All subscription SKUs that this app offered that are present in the transactions data were found to have a zero or negative pass-through rate in Williams' analysis.

[2] But-for list prices are calculated as the actual list price minus the but-for dollar reduction in the service fee multiplied by the pass-through rate for the respective month, app, monetization type, and app category, as estimated by Singer. The dollar reduction in the service fee is calculated as the list price observed in Google's transactions data times difference between the service fee rate observed in Google's transactions data and the but-for service fee rate for subscriptions according to Singer's Table 5.

Sources:

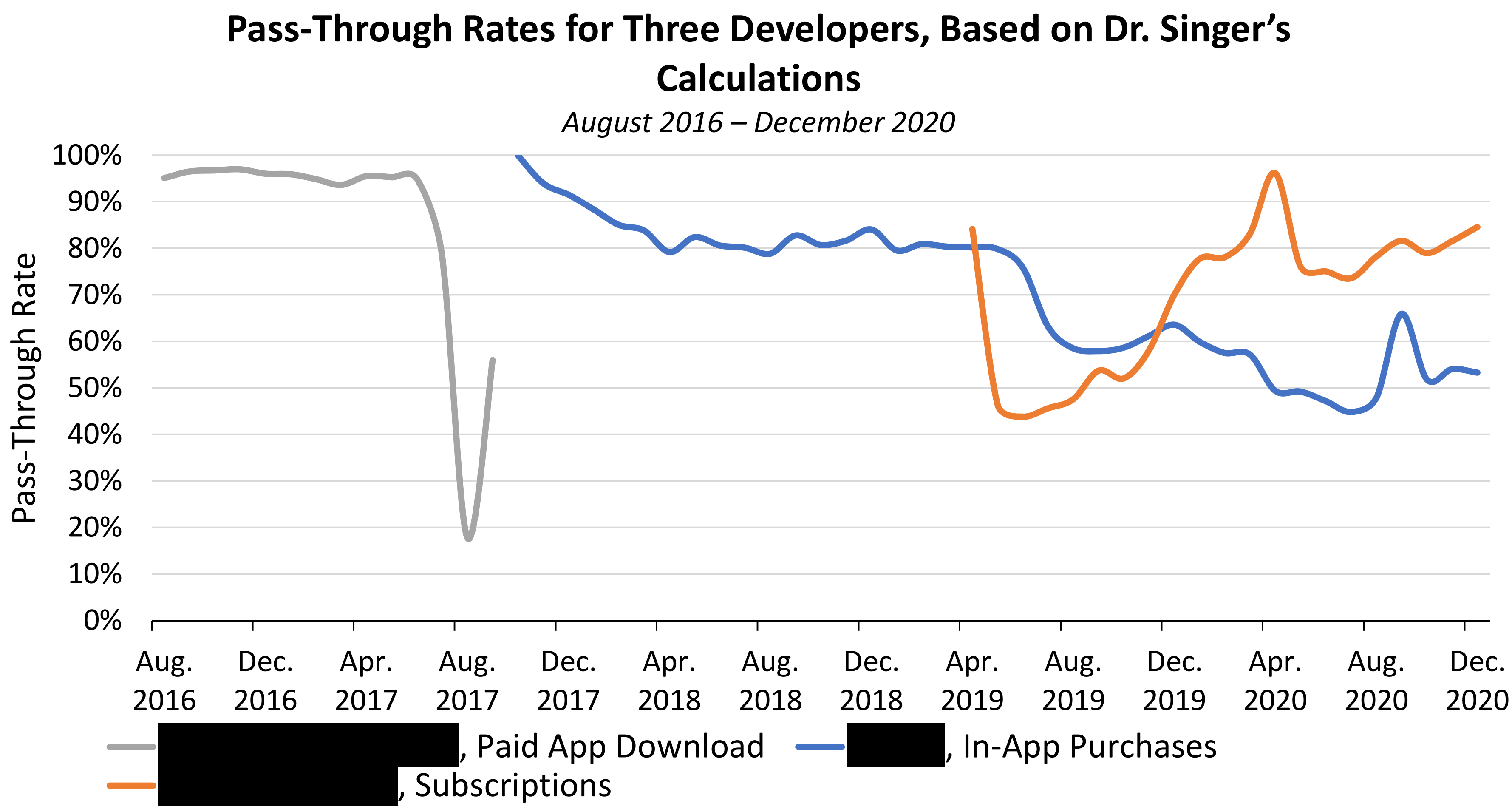
[1] Singer backup for Tables 5, 7, & 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886

[3] Google Transactions Data, GOOG-PLAY-007203251

[4] Williams backup for Figure 3

Exhibit 58



Note:

[1] This graph shows Dr. Singer's monthly pass-through rates for paid app downloads from "[redacted]", non-subscription IAPs from "[redacted]", and subscription IAPs from "[redacted]" for August 2016 through December 2020.

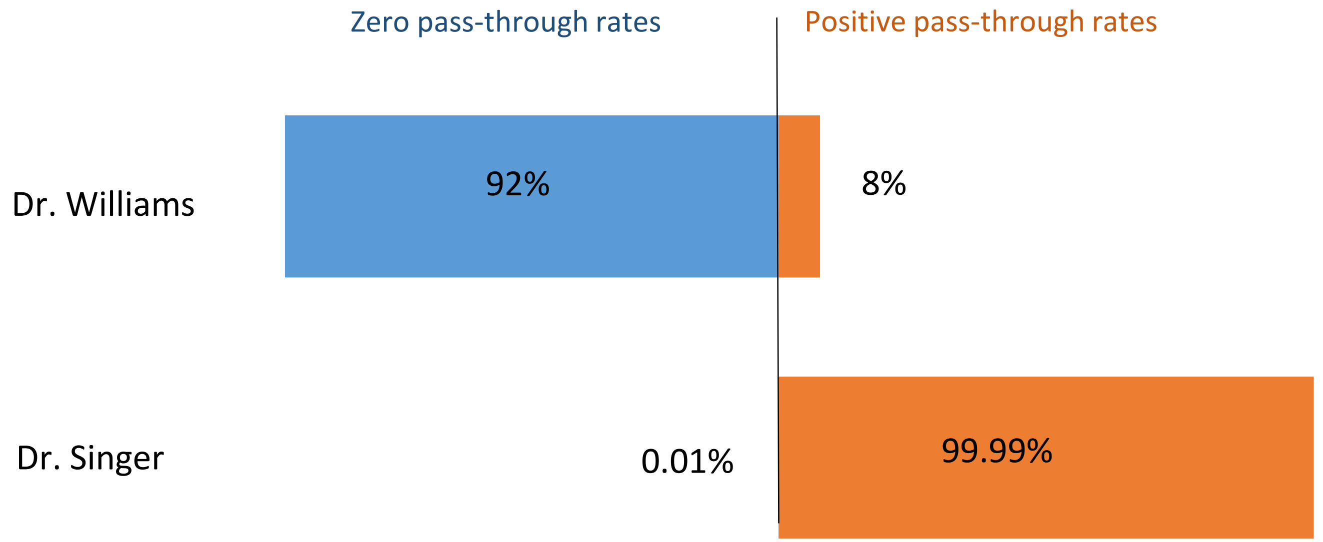
Sources:

[1] Singer backup for Tables 7 and 8

[2] App-level spend data, Consumers, GOOG-PLAY-005535886

Exhibit 59

Shares of Positive Pass-Through Rates Found by Dr. Singer and Dr. Williams Are Nearly Opposite of Each Other



Notes:
[1] Zero pass-through rates combine zero and negative pass-through rates.
[2] Dr. Williams' pass-through rates are calculated at the product ID and monetization type level, whereas Dr. Singer's pass-through rates are calculated at the developer ID, app category, monetization type, and year-month level.

Sources:
[1] Singer backup for Tables 7 & 8
[2] Williams Figure 3

Exhibit 60

U.S. Consumers' Play Points Earned Transactions

Among Only Transactions with Earned Points

	Among Transactions with Earned Play Points					Among all U.S. Consumer Transactions	
Year	Number of Transactions	Number of U.S. Consumers	Number of Developers	Number of Apps	Gross Consumer Spend	Class Consumer Counts by Year	Percent of Consumer Class Earning Points
Nov 2019 - Dec 2019	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
2020	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
2021	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Starting November 2019	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Notes:

[1] The Play Points program began in November 2019 in the U.S. The transactions data shows transactions until July 3rd, 2021.
[2] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.
[3] Gross Consumer Spend is the list price which is determined by pre-tax price per unit.
[4] NULL point transactions are seen as consumers not enrolled. 0 points earned transactions are seen as enrolled consumers not earning points. NULL point transactions are excluded but 0 points earned transactions are included.

Sources:

[1] Google Transactions Data, GOOG-PLAY-007203251
[2] "Google Play Points: a rewards program for all the ways you Play," https://blog.google/products/google-play/google-play-points-rewards-program-all-ways-you-play/

Exhibit 61

Upper vs Lower Bound of U.S. Consumers' Play Points Redeemed Transactions

Starting November 2019	Number of Redeeming U.S. Consumers	Redeeming U.S. Consumers as Percent of U.S. Consumers Earning Points	Redeeming U.S. Consumers as Percent of U.S. Consumer Class
Lower Bound: Redemptions recorded in data	[redacted]	[redacted]	[redacted]
Upper Bound: Redemptions recorded in data or inferred from Stored Value	[redacted]	[redacted]	[redacted]

Notes:

[1] The Play Points program began in November 2019 in the U.S. The transactions data shows transactions until July 3rd, 2021.

[2] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

[3] Gross Consumer Spend is the list price which is determined by pre-tax price per unit.

[4] The upper bound for counts of consumers with redeemed Points is determined by consumers earning at least 10 lifetime Play Points and having at least one Play Credit transaction on or after November 2019. It assumes that Play Credit was earned through Play Points if not recorded in the data as a Points redemption.

[5] Some consumers in the lower bound also make stored value purchases.

[6] Play Point earners can redeem for Google Play Credit to be used on movies, books, etc. which would not show up in the transactions data.

Source:

[1] Google Transactions Data, GOOG-PLAY-007203251

Exhibit 62

Developers with Conflicting Country Locations in App-Level Spend Data and App Catalog

August 2016 - December 2021

	Count of Developers	% of Developers
"U.S." Developers with some but not all apps identified as non-U.S. in Catalog	140	0.3%
"U.S." Developers with all apps identified as non-U.S. in Catalog	3,632	7%
"U.S." Developers with all apps missing Catalog location	5,836	12%
"U.S." Developers with matching Catalog location	39,779	81%
Total	**49,387**	**100%**

Notes:

[1] "U.S." Developers with some but not all apps identified as non-U.S. in Catalog Data includes 111 developers who have some missing locations in the App Catalog, but who are identified as non-U.S. in at least one app with a non-missing Catalog location.

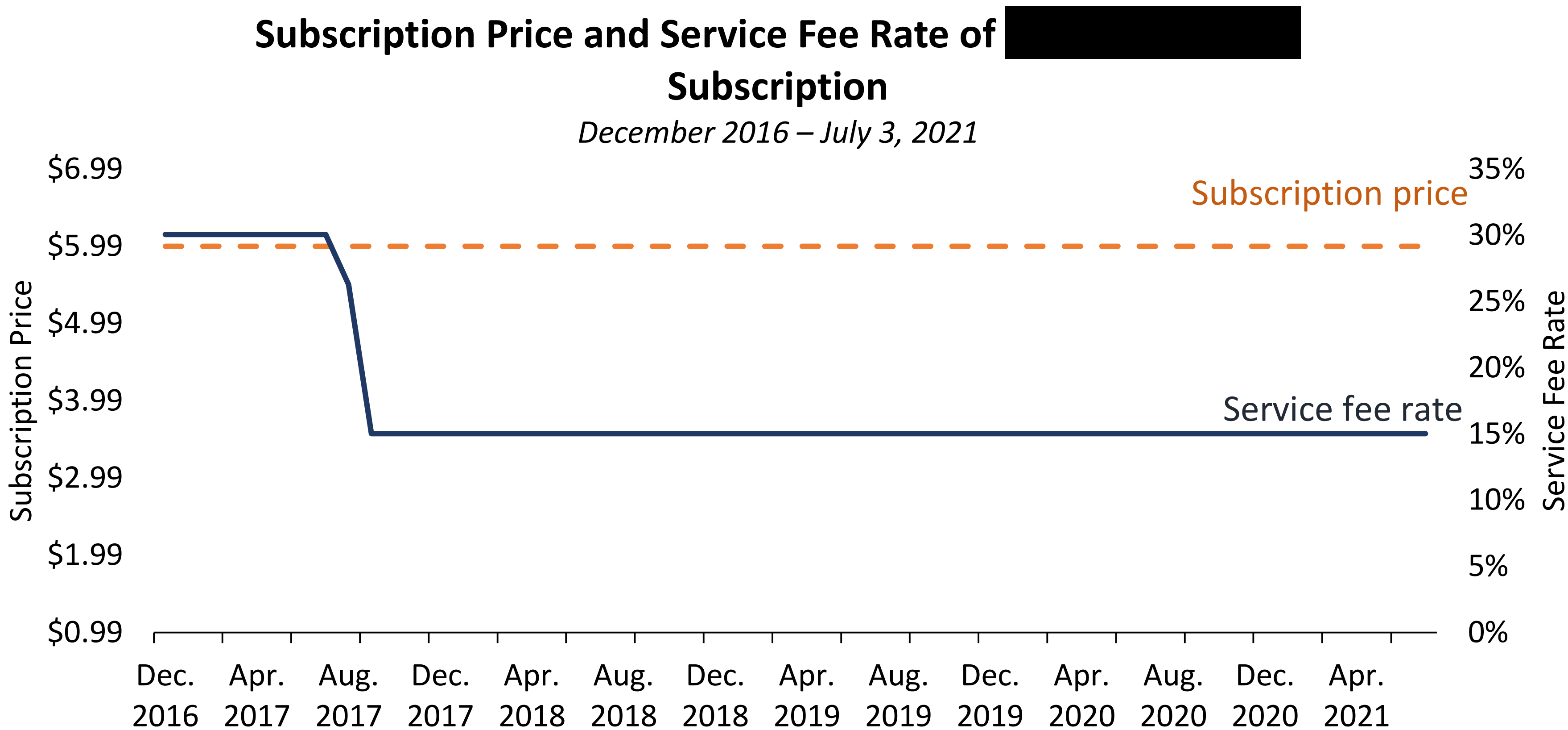
[2] "U.S." Developers with matching Catalog location includes 616 developers who have some missing locations in the Catalog data, but who are identified as U.S. in all apps with a non-missing Catalog location.

Sources:

[1] App-level spend data, Developers, GOOG-PLAY-005535885 and GOOG-PLAY-010801689

[2] Play App Catalog, GOOG-PLAY-001507601 and GOOG-PLAY-001507602

Exhibit 63



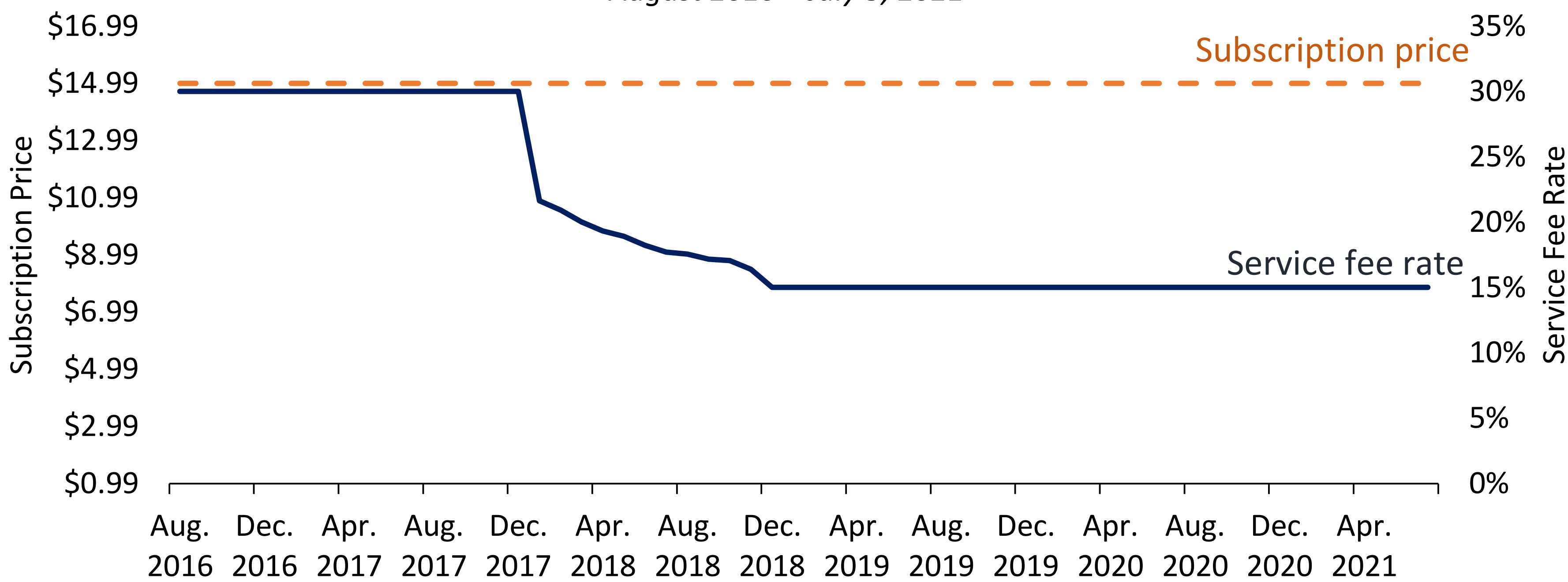
Notes:
[1] This graph shows monthly prices and service fee rates for SKU "[redacted].
[2] Both Burtis and Williams found zero pass-through for this SKU. Singer's predicted pass-through rates for this subscription SKU from this developer in the "Music and Audio" app category range from 80.6% to 99.999% from August 2016 to December 2020.
[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Source:
[1] See Exhibit 52 for sources.

Exhibit 64

Subscription Price and Service Fee Rate of Premium Digital [redacted] Subscription

August 2016 – July 3, 2021



Notes:
[1] This graph shows monthly prices and service fee rates for SKU "[redacted]".
[2] Both Burtis and Williams found zero pass-through for this SKU. Singer's predicted pass-through rates for this subscription SKU from this developer in the "News and Magazines" app category range from 71.1% to 94.3% from August 2016 to December 2020.
[3] Transactions are charged, delivered, in USD, not refunds and pre-tax price per unit greater than $0. Test instruments are excluded.

Source:
[1] See Exhibit 52 for sources.